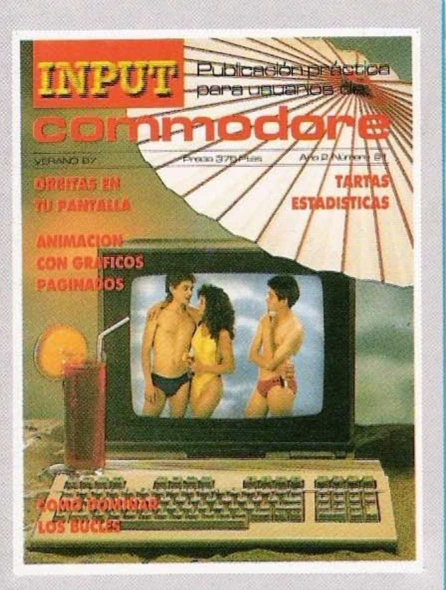

INPUT
Publicación práctica para usuarios de
commodore
VERANO 87
Precio 375 Ptas
Año 2 Número 21
ORBITAS EN TU PANTALLA
TARTAS ESTADISTICAS
ANIMACION CON GRAFICOS PAGINADOS
COMO DOMINAR LOS BUCLES



AÑO 2 NUMERO 21

DIRECTOR: Manuel Pérez
DIRECTOR DE ARTE: Luis F. Balaguer
REALIZACION GRAFICA: Didac Tudela
COLABORADORES: Antonio Pliego, Xavier Ferrer, Josep M.ª Gils, Christophe Pais, Jaime Mardones, Equipo Molisoft, Carles Bartolomé, Ramón Ollé, Angels Alvarez
FOTOGRAFIA: Ernesto Wallfisch, Joan Boada

INPUT Commodore es una publicación de PLANETA-DE AGOSTINI, S.A.
GERENTE DIVISION DE REVISTAS:
Sebastián Martínez

PUBLICIDAD: José Real-Grupo Jota
Madrid: c/ General Varela, 35
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.º
Teléf. 250 23 99

FOTOMECANICA: TECFA, S.A.

IMPRESION: Sirven Grafic
c/ Gran Vía, 754-756, 08013 Barcelona
Depósito legal: B. 38.114-1986

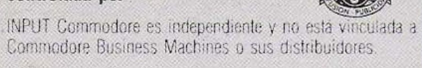
SUSCRIPCIONES: EDISA
López de Hoyos, 141. 28002 Madrid
Teléf. (91) 415 97 12

REDACCION:
Aribau, 185, 1.º
08021 Barcelona

DISTRIBUIDORA:
R.B.A. PROMOTORA DE EDICIONES, S.A.
Calle B, n.º 11. Sector B. Zona Franca
08004 Barcelona

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT Commodore es una publicación controlada por



INPUT commodore

SUMARIO



VACACIONES

Se acercan las vacaciones de verano y las noches estivales se prevén largas, cálidas y fantasiosas. Los ordenadores serán el centro de atención de todas las noctámbulas miradas, convirtiendo los aburridos mitines/veladas en fiestas multicolores de sonidos, música y explosiones.

Poco a poco, los joysticks ocupan sus puertos/puestos, el cansancio se va adueñando de nuestras muñecas y los reflejos decaen al paso del tiempo. Llega el final y las despedidas, regresando el silencio a la paz del hogar, a la pantalla del monitor.

El ordenador, compañero de desventuras alienígenas, descansa de sus batallas galácticas interestelares. Las luces se apagan y una nueva etapa comienza, más intimista, más satisfactoria y provechosa: llegar al corazón del microprocesador, seducir al programa residente en memoria, descubrir sus interioridades... Eso es lo que podríamos llamar seducción de un programa.

La noche es un buen estadio para conseguir records, olímpicos o mentales («mens sana in corpore sano»), un circuito donde correr contra reloj contra otros programadores, un «campus» de soledad y duelo entre la máquina y el hombre.

En esas horas, el BASIC se convierte en delator y el código máquina en enemigo. Las sentencias chocan contra los errores sintácticos y los desensambladores son paladines de causas aun por defender. La lucha es ardua, cruel, salvaje,... los párpados sucumben al sueño, la mente obnubilada es presa de ciegas fantasías y lo que creíamos poder hacer se convierte en utopía...

Una nueva mañana clarea en el horizonte y el RETURN del cansancio nos RESETEA hacia otra SUBRUTINA: la del sueño.

DIAGRAMAS DE BARRAS

- UN DIAGRAMA DE BARRAS A TODO COLOR
- GENERALIDADES
- UTILIZACION DEL PROGRAMA
- INTRODUCCION DE LA INFORMACION
- CORRECCION DE LOS DATOS
- TRAZADO DEL GRAFICO
- ESCALADO DE LOS EJES

Para efectuar un análisis instantáneo facilita las cifras y, olvidándote de las matemáticas, observa cómo tu micro las convierte en un diagrama de barras a todo color y con un aspecto totalmente profesional.

Todo el mundo habrá visto referencias de los sistemas informáticos de oficina que muestran elaboradas representaciones de datos tales como cifras de compras e inventarios. Esta clase de información, cantidad de cifras virtualmente incomprensibles, se presenta mejor en forma de gráfico o de tabla, que se comprende más fácilmente de una ojeada. Prepararlo a mano es tedioso, pero precisamente es el tipo de trabajo en el que destacan los ordenadores de oficina, con su capacidad para dar una presentación a todo color en cuestión de segundos.

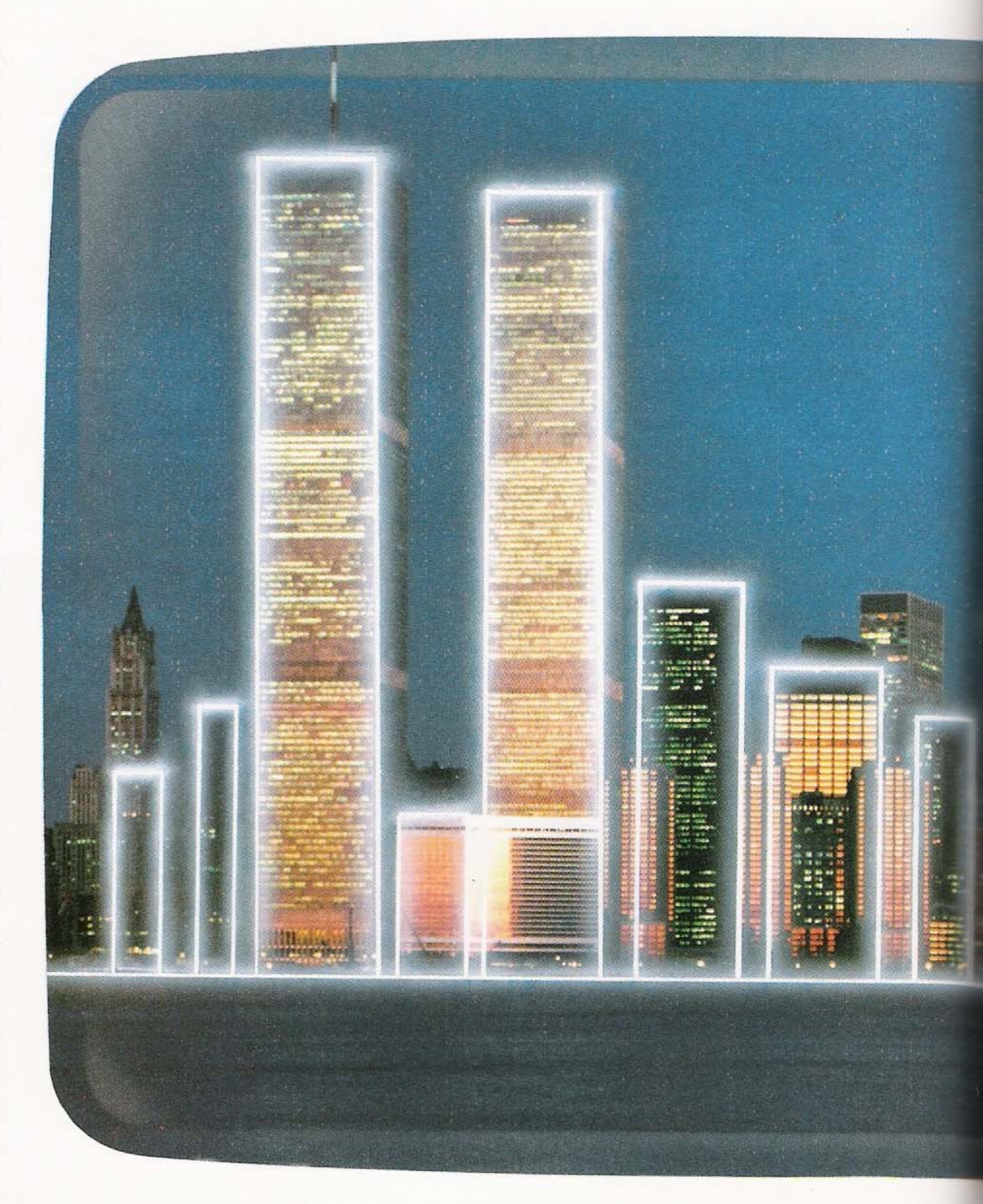
La presentación de dichos datos no sólo se limita a las máquinas de oficina, sino que también es una de las aplicaciones prácticas de un ordenador doméstico.



No es probable que el usuario medio de dicho tipo de ordenadores maneje algo parecido a la vasta cantidad de información generada incluso por un negocio pequeño, pero hay muchas cosas que pueden analizarse útilmente en un ordenador. Por ejemplo, ¿se incrementaron los ingresos y los gastos hacia fin de año, o disminuyeron los gastos? ¿Has gastado más en software informático (o en otros artículos tales como revistas, entretenimiento y consumo) durante el otoño que durante el verano? ¿Cuándo sobrepasan tus ahorros un cierto nivel y durante cuánto tiempo permanecen así?

Junto a dichas aplicaciones de tipo comercial hay toda clase de temas de interés general, quizás datos y cifras relacionadas con un hobby, que podrían ser analizados y representados. Dichos temas van, por ejemplo, desde las cifras de asistencia en el club local hasta los niveles de precipitación o de las mareas. Otras cosas que puedes desear representar gráficamente son resultados deportivos, o el tamaño de una colección en crecimiento.

El sencillo programa ofrecido aquí te permite preparar rápidamente una representación visual de cualquier estadística que varíe a lo largo de un período de tiempo. Los ejes del gráfico se ajustan automáticamente, por lo que puedes entrar cifras semanales, mensuales o anuales. De hecho, puedes utilizar cualquier unidad de tiempo.



Puedes manejar valores que lleguen a cualquier magnitud, unidades, decenas, centenas, millares o incluso millones, si tus cálculos llegan tan lejos.

UTILIZACION DEL PROGRAMA

Al lanzar el programa, visualiza un «menú» o lista de opciones. Seleccionando la opción de introducir nuevos datos, se te invita a indicar los nombres de los ejes. Los rótulos que teclees aparecerán en el gráfico cuando se dibuje el mismo. Cuando introduzcas dichos rótulos, recuerda hacerlo correctamente. El número de barras representando semanas, meses, años o lo que sea, se representa a lo largo del eje x, mientras que el valor de las barras, pesetas o cualquier otra unidad, se representa a lo largo del eje y.

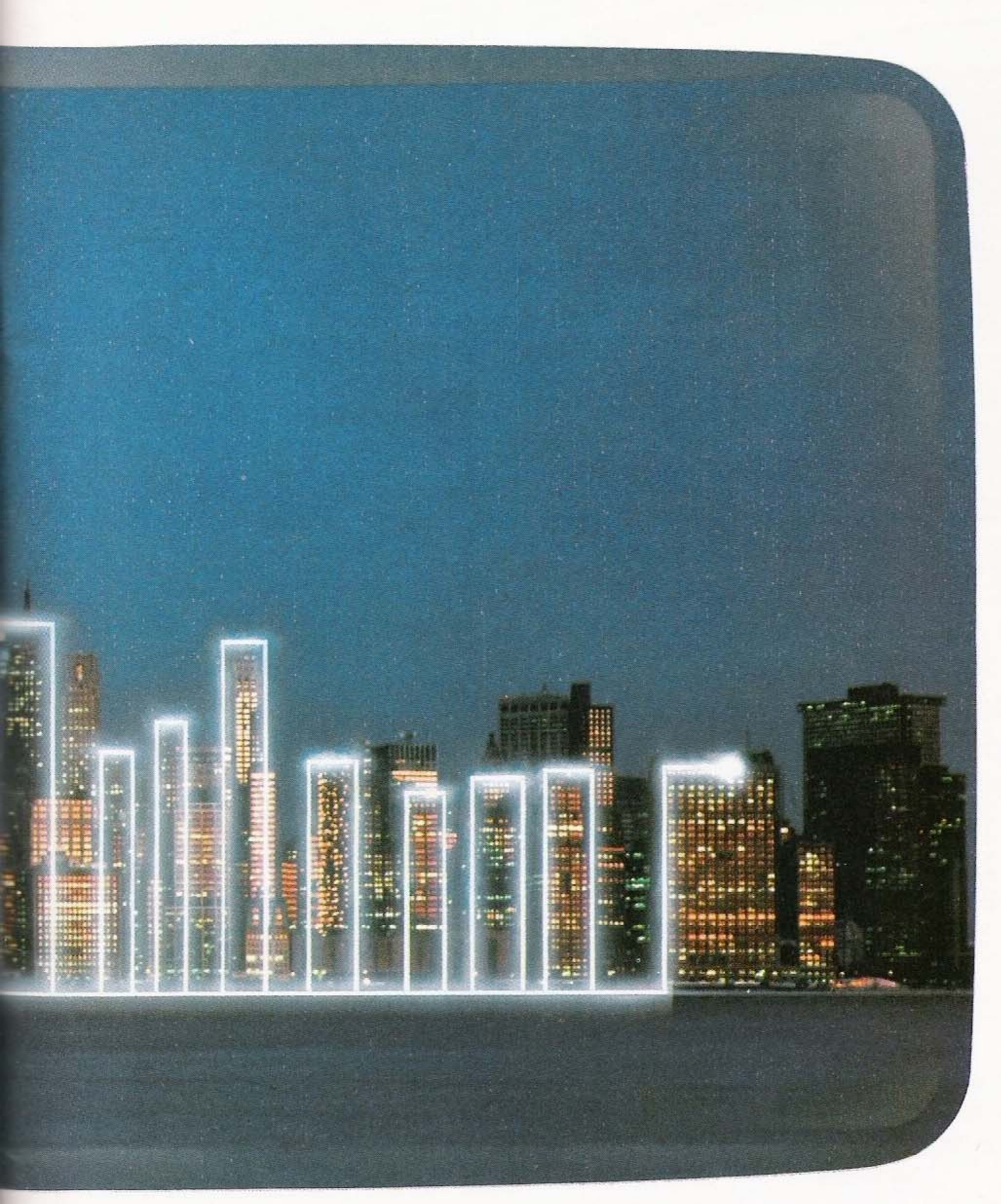
A continuación se te pide que introduzcas el número de barras a trazar. Puedes trazar un máximo de 30 barras.

Luego te pregunta los datos. Imprime el número de cada barra y te pide que introduzcas el valor, que puede ser positivo o negativo. Dichos valores van de -1038 a 1038.

Una vez introducido el último valor, se vuelve al menú de modo que puedas elegir o bien corregir los datos, modificando los valores introducidos en caso de que hayas cometido algún error, o bien trazar el gráfico. Si eliges corregirlos, los valores se imprimirán correlativamente en la pantalla. Debes pulsar cualquier tecla excepto la tecla RETURN para dejar sin cambios el valor y visualizar el siguiente, o pulsar la tecla RETURN e introducir a continuación el nuevo valor.

Cuando estés satisfecho con los valores, selecciona la opción de visualizar o trazar el gráfico.

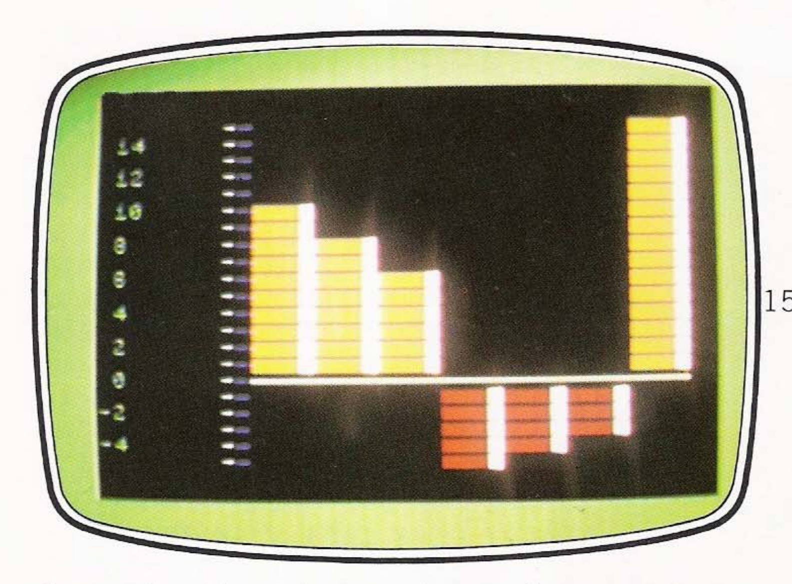
Puedes elegir entre un gráfico escalado o un gráfico a escala completa. La opción de gráfico escalado visualiza el gráfico con las dimensiones correspondientes al eje y redondeadas a un máximo de diez, cien, mil, etc. según el valor máximo de los datos, pero el gráfico no llena la totalidad de la pantalla. La opción de pantalla entera



visualiza el gráfico sobre toda el área de la pantalla del televisor, pero imprime los valores reales de los datos (no los valores escalados) a lo largo del eje y.

Para mayor claridad, las barras trazadas van separadas por un pequeño espacio o por una barra coloreada.

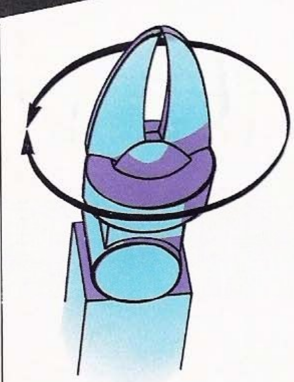
```
Ø BB=53280:GOTO 2ØØ
1 CLR:BB=5328Ø:FF=1
2 POKE BB,Ø:POKE BB+1,Ø
3 Q$="[ 25*CRSR ABAJO]"
4 INPUT "[SHIFT+CLR/
  HOME]NOMBRE DE LA
  GRAFICA"; N$
5 INPUT "SHIFT+CLR/
  HOME][CTRL+8]NUMERO
  DE BARRAS.(1-3Ø)"; XX
6 IF XX<1 OR XX>3Ø THEN5
7 DIM A(XX),B(XX),C(XX)
8 FOR Z= TO 3Ø
9 IF XX*Z>3Ø THENX1=Z-
  1:GOTO 11
1Ø NEXT Z
11 PRINT "[SHIFT+CLR/
   HOME]ENTRAR DATOS
   BARRAS:[CRSR ABAJO]"
12 FOR Z=1 TO XX:PRINT
   "BARRA N."; Z;:INPUT
   C(Z):NEXT Z
13 A=Ø:B=Ø:C=Ø:D=Ø
14 FOR Z=1 TO XX
15 IF C(Z)>A THENA=C(Z)
16 IF C(Z)<B THENB=C(Z)
17 IF C(Z)<Ø
   THENB(Z)=C(Z):A(Z)=Ø
18 IF C(Z)>Ø
   THENA(Z)=C(Z):B(Z)=Ø
19 NEXT Z
2Ø C=A+ABS(B)
22 GOTO 2ØØ
24 POKE BB,5:POKE BB+1,Ø
25 PRINT "[SHIFT+CLR/
   HOME]"N$
26 FOR Z=2 TO 22
27 POKE 1Ø33+Z*4Ø,64:POKE
   553Ø5+Z*4Ø,6
28 POKE 1Ø32+Z*4Ø,45:POKE
   553Ø4+Z*4Ø,3
29 NEXT Z
3Ø FOR Z=1 TO XX:W=Ø:CO=8
31 IF B(Z)<Ø THENCO=2
32 FOR ZZ=B/D TO A/D
33 W=W+1:IF INT (ZZ)<>Ø
   THEN38
34 FOR G=1 TO X1
35 POKE (1953-
   X1)+(Z*X1+G)–W*4Ø,64
36 POKE (56225-
   X1)+(Z*X1+G)–W*4Ø,7
37 NEXT G:GOTO 43
38 FOR G=1 TO
   X1:CA=227:CC=CO:IF G<>
   1 AND G=X1 THENCO=1
39 IF ZZ<Ø THENCA=228
4Ø IF B(Z)/D>ZZ OR A(Z)/D<ZZ
   THEN43
41 POKE ([1953-
   X1]+Z*X1+G)–W*4Ø:CA:POKE
   ((56225-
   X1)+Z*X1+G)–W*4Ø,CO
42 CO=CC:NEXT G
43 POKE BB:RND(1)*2+5
44 NEXT ZZ,Z:POKE BB,5
45 PRINT "[CLR/
   HOME][CTRL+6][CRSR
   ABAJO]"; LEFT$(Q$,2Ø-
   ABS(B/D))
46 FOR Z=Ø TO A STEP D*2:IF
   C=<1Ø AND INT (Z)<>Z
   THEN PRINT "[ 3*CRSR
   ARRIBA]":GOTO 48
47 PRINT INT (Z)"[ 3*CRSR
   ARRIBA]"
```



Commodore 64: cada barra produce un efecto.

```
48 NEXT Z
49 PRINT "[CLR/HOME][CRSR ABAJO]"; LEFT$(Q$,2Ø-ABS(B/D))
5Ø FOR Z=Ø TO B STEP-D*2:IF C=<1Ø AND INT (Z)<>Z THEN PRINT "[CRSR ABAJO]":GOTO52
51 PRINT INT (Z) "[CRSR ABAJO]"
52 NEXT Z
53 POKE 198,Ø
54 GET A$:IF A$="" THEN54
55 GOTO 2ØØ
1ØØ POKE BB,1:POKE BB+1,1
1Ø3 IF FF=Ø THEN5ØØ
1Ø5 PRINT "[SHIFT+CLR/HOME] [CTRL+5] [CTRL+9] "TAB(1Ø) "DIBUJO GRAFICO DE BARRAS"
11Ø PRINT "[2*CRSR ABAJO] [ 3 ESPACIOS] [CTRL+6] ↑ "
115 FOR Z=1 TO 15
12Ø PRINT "[ 3 ESPACIOS][SHIFT+B] ":NEXT Z
125 PRINT"[ 3 ESPACIOS][CTRL+3] [SHIFT+W] [CTRL+4)",
13Ø FOR Z=1 TO 3Ø
135 PRINT ""–""; :NEXT Z
14Ø PRINT ">"
145 PRINT"[CLR/HOME][CTRL+3][7*CRSR ABAJO][5*CRSR DCHA.] VALOR: UNIDADES ENTRADAS"
15Ø PRINT "[CTRL+7][1Ø*CRSR ABAJO][5*CRSR DCHA.] BARRAS: DIAS/ SEMANAS/ MESES"
155 PRINT "[3*CRSR ABAJO][CTRL+5][CTRL+9] "TAB
16Ø 5)"PULSAUNATECLAPARA SEGUIR"
16Ø POKE198,Ø
165 WAIT198,1
17Ø GOTO24
2ØØ POKEBB,6:POKEBB+1,6
21Ø PRINT"[SHIFT+CLR/HOME][7*CRSRABAJO]"TAB(15)"[CTRL+4][CTRL+9][2 ESPACIOSOPCION][2 ESPACIOS][CRSR ABAJO][CTRL+2]"
22Ø PRINTTAB(12)"1ENTRAR DATOS"
23Ø PRINTTAB(12)"2GRAFICOS AESCALA"
24Ø PRINTTAB(12)"3VER/ EDITARDATOS"
245 PRINTTAB(12)"4GRAFICOS AMPLIADOS"
25Ø PRINTTAB(13)"[CRSR ABAJO][CTRL+4][CTRL+9] ESCOGERFUNCION?"
26Ø POKE198,Ø
27Ø GETA$
28Ø ONVAL(A$)GOTO4ØØ,7ØØ, 3ØØ,6ØØ
29Ø GOTO27Ø
3ØØ IFFF=ØTHEN5ØØ
3Ø5 POKEBB,7:POKEBB+1,7
31Ø PRINT"[SHIFT+CLR/HOME][CTRL+5]"N$" DATOS:"
315 PRINT,"[CRSR ABAJO][CTRL+3]BARRA", "[CTRL+7]ALTURA[CRSR ABAJO]"
32Ø FORZØ1TOXX
325 IFA(Z)<>ØTHENPRINT, "[CTRL+7]"Z, "[CTRL+3]"A(Z):GOTO335
33Ø PRINT,"[CTRL+3]"Z, "[CTRL+7]"B(Z)
335 POKE198,Ø
34Ø GETA$
345 IFA$=""THEN34Ø
35Ø IFA$<>CHR$(13)THEN 37Ø
355 PRINT"[CRSR ARRIBA][CTRL+1]"TAB(18);
36Ø INPUTC(Z)
365 GOTO13
37Ø NEXTZ
375 GOTO2ØØ
4ØØ POKEBB,13:POKEBB+1,13
41Ø PRINT"[SHIFT+CLR/HOME][8*CRSRABAJO][6*CRSR DCHA.][CTRL+6] PULSAR[CRSRDCHA.] ([CTRL+7]E[CTRL+6]) NTRARDATOS,([CTRL+7]M [CTRL+6])ENU"
42Ø GETA$
43Ø IFA$="E"THEN1
44Ø IFA$="M"THEN2ØØ
45Ø GOTO42Ø
5ØØ PRINT"[SHIFT+CLR/HOME][CTRL+1]"TAB(16) "NOHAYDATOS!"
51Ø FORZ=1TO5ØØ:NEXTZ
52Ø GOTO2ØØ
6ØØ IFFF=ØTHEN5ØØ
6Ø5 IFC<=1ØTHEND=.5:GOTO 1ØØ
6Ø6 IFC<=2ØTHEND=1:GOTO 1ØØ
61Ø D=C/2Ø:GOTO1ØØ
7ØØ IFFF=ØTHEN5ØØ
7Ø5 IFC<=1Ø THEND=1.25:GOTO1ØØ
7Ø6 IFC<=2ØTHEND=2.5:GOTO 1ØØ
71Ø D=C/2Ø+9:D=D*.1:D=INT (D)*1Ø:GOTO1ØØ
```





En INPUT estamos convencidos de que aún puedes hacer muchas más cosas con tu ordenador. Sin duda, muchos lectores estareis utilizando vuestro micro para funciones de lo más variadas, en unos casos; pintorescas, en otros; mientras que algunos listillos habrán podido utilizarlo para resolver tareas complejas. Es lógico, modificando programas y variando los periféricos nuestro ordenador puede prestar sus servicios en infinidad de facetas. INPUT quiere que esas aplicaciones y utilidades a las que has conseguido dedicar tu ordenador, sean conocidas por todos sus lectores y por eso ha organizado el «Concurso de Aplicaciones y Utilidades», en el que puede participar cualquiera de nuestros lectores.





UTILIDADES Y APLICACIONES: Si tu ordenador controla la calefacción de tu casa, gobierna un robot, dirige un pequeño negocio, organiza la maqueta de tu tren eléctrico, o cualquier cosa interesante u original; envíanos información gráfica y listados de tus programas, grabados en un cassette, diskette o microdrive.

Todo ello habrá de venir acompañado por un texto que aclare cuál es su objetivo, el modo de funcionamiento y una explicación del cometido que cumplen las distintas rutinas que lo componen. El texto se presentará en papel de tamaño folio y mecanografiado a dos espacios. No importa que la redacción no sea muy clara y cuidada; nuestro equipo de expertos se encargará de proporcionarle la forma más atractiva posible.

UN JURADO propio decidirá en cada momento qué colaboraciones reúnen los requisitos adecuados para su publicación, y evaluará la cuantía del premio en metálico al que se hagan acreedoras.

No olvideis indicar claramente para qué ordenador está preparado el material, así como vuestro nombre y dirección y, cuando sea posible, un teléfono de contacto. Entre todos los trabajos recibidos durante los próximos tres meses SORTEAREMOS:

- **Un premio de 50.000 ptas.**
- **Un premio de 25.000 ptas.**
- **Un premio de 10.000 ptas.**
  en material microinformático a elegir por los afortunados.

¡No os desanimeis!, por muy simples o complejas que puedan parecer vuestras ideas, todas están revisadas con el máximo interés.

INPUT COMMODORE
Aribau, 185. Planta 1.ª
08021 BARCELONA

NOTA: INPUT no se responsabiliza de la devolución del material que no vaya acompañado por un sobre adecuado con el franqueo correspondiente.

GRAFICOS DEFINIDOS POR EL USUARIO (II)

He aquí dos nuevos personajes —una serpiente y un mono— para completar el escenario de la jungla, más algunas ideas para animar el cuadro y para ahorrar memoria.

En los capítulos anteriores de esta serie habrás visto la manera de definir una gran cantidad de figuras en tu ordenador y has empezado a formar un cuadro de una escena de la jungla para utilizarlas.

El programa de la escena de la jungla está formado por secciones, en las que se definen en cada una los UDG. Cada sección forma parte del cuadro —el cocodrilo, el elefante, los árboles, etc.—. La segunda parte del programa que sigue añade una serpiente y un mono y termina el fondo.

Si has almacenado la primera parte del programa en cinta o en diskette, podrás cargarlo en el ordenador, porque este programa requiere la primera parte para funcionar.

Las líneas adicionales de datos que necesitas para definir las imágenes se escriben después del programa principal. En primer lugar escribe estas líneas:

```
51 FOR Z=144 TO 759:READ
   X:POKE 13312+Z,X:NEXT Z
1ØØ M$="[CRSR
   ABAJO][CTRL+9][ 2*CRSR
   DCHA.]/Ø[ 2*CRSR
   DCHA.]12[CRSR ARRIBA].[
   2*CRSR ABAJO][ 9*CRSR
   IZQDA.]"
11Ø M$=M$+"[CTRL+9][CRS
   R DCHA.]345[CRSR DCH
   A.]67[CRSR ABAJO] [ 6*
   CRSR IZQDA.]89[ 2*CRS
   R DCHA.]:;<[CRSR ABA
   JO][ 6*CRSR IZQDA.]=>[
   CRSR DCHA.]?[SHIFT+*][
   SHIFT+A][SHIFT+B][CRS
   R ABAJO][ 7*CRSR IZQD
   A.][SHIFT+C][SHIFT+D][S
   HIFT+E][SHIFT+F][SHIFT
   +G][SHIFT+H][SHIFT+I][
   SHIFT+J][CRSR ABAJO]
   [ 8*CRSR IZQDA.]"
12Ø M$=M$+"[CTRL+9][SHIF
   T+K][ 4*SHIFT+L][SHIFT
   +M][SHIFT+N][SHIFT+O]
   [CRSR ABAJO][ 9*CRSR I
   ZQDA.][SHIFT+P][SHIFT+
   Q][SHIFT+R][ 2*CRSR D
   CHA.][SHIFT+S][SHIFT+T
   ][SHIFT+U][CRSR ABAJO]
   [ 9*CRSR IZQDA.][SHIFT
   +V][SHIFT+W][SHIFT+X][
   SHIFT+Y][ 2*CRSR DCHA
   .][SHIFT+Z][SHIFT+ +][C
   RSR ABAJO][ 6*CRSR IZ
   QDA.][COMM+ −][SHIFT
   + −][CRSR ABAJO][ 2*C
   RSR IZQDA.][SHIFT+ ↑][
   CTRL+4]"
14Ø PRINT "[ 3*CRSR ABAJO
   ][CTRL+2][ 4*CRSR DCH
   A.][CTRL+9]RUX[↑!"C$
   "[ 4*CRSR DCHA.][CTRL
   +9]SVY←"CHR$(34) "$&(
   *,"C$"[ 4*CRSR DCHA.][
   CTRL+9]TWZ] #%')+−"
155 POKE 1Ø88,81
175 TT=T:PRINT
   "[CLR/HOME]"TAB(T)
   "[CTRL+3]"M$:FOR D=1
   TO 5Ø:POKE 5536Ø,
   RND(1)* 7+1:NEXT
   D
18Ø PRINT "[CLR/HOME]"TAB(T)
   "[CTRL+1]"M$
```

A continuación escribe estas líneas de datos para definir las formas de los UDG.

```
4ØØ REM SERPIENTE
41Ø DATA Ø,Ø,Ø,Ø,Ø,Ø,3,14,31,
   31,63,67,65,254,254,22
   5,71,63,Ø,Ø,Ø,Ø,Ø,Ø,Ø
42Ø DATA Ø,Ø,Ø,Ø,255,12,28,2
   8,28,254,143,1,Ø,4,248,
   2Ø8,224
143Ø DATA Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
   Ø,128,24Ø,56
144Ø DATA 52,99,225,225,225,
   96,47,31,14,6,2,1,Ø,Ø,Ø
   ,Ø,Ø,Ø,Ø,Ø,3,5,63,
   127
145Ø DATA 255,225,24Ø,248,2
   52,255,255,126,6Ø,28,
   124,247,247,115,31,1,
   Ø,Ø,Ø,Ø
146Ø DATA 255,191,159,143,1
   99,194,127,64,128,128
   ,64,63
147Ø DATA 48,112,248,249,25
   3,255,189,255,Ø,Ø,Ø,Ø,
   192,252,19Ø,63,63,24,
   24Ø
148Ø DATA 9,5,7,5,249,67,243
   ,251,255,252,238,239,
   255
149Ø DATA 192,224,2Ø8,223,2
   24,129,193,227,243,24
```



- DATOS PARA COMPLETAR LA ESCENA DE LA JUNGLA
- UNA SERPIENTE Y UN MONO
- COMPLETANDO EL FONDO
- EMPLEO DE GRAFICOS DE ALTA RESOLUCION
- ANIMACION DE LOS CARACTERES
- ALMACENAMIENTO DE DATOS EN CODIGO MAQUINA

```
     7,255,248,24Ø,224,
     192
15ØØ DATA Ø,Ø,Ø,Ø,255,129,19
     5,199,199
151Ø DATA 239,239,255,Ø,Ø,Ø,
     Ø,Ø,Ø,Ø,Ø,128,248,31,1
     4Ø,158,191,191,255,Ø,
     Ø,Ø,Ø
152Ø DATA Ø,Ø,Ø,1,15,62,2Ø6,
     3Ø,61,121,243,247,247
     ,63,15,3,Ø
153Ø DATA 3Ø,127,191,95,93,
     61,25Ø,24Ø,194,252,19
     4,192,192,224,224,24Ø
154Ø REM MONO
155Ø DATA 1,2,4,8,16,32,64,1
     28,Ø,Ø,Ø,3,31,63,127,2
     52,Ø,Ø,Ø,224,248,252,2
     52
156Ø DATA 62,3,15,15,15,15,1
     4,14,12
157Ø DATA 3,142,24Ø,192,128,
     Ø,Ø,Ø,1,1,3,3,3,3,3,3,2
     48,24Ø,224,224,
     192
158Ø DATA 192,192,192,3Ø,14,
     11Ø,76,56,Ø,Ø,Ø,Ø,Ø,Ø,Ø
     ,Ø,Ø,Ø
159Ø DATA 1,28,28,56,56,112,
     112,224,224
16ØØ DATA 3,3,1,1,1,Ø,Ø,Ø,19
     2,224,224,24Ø,248,248
     ,252,126,3,7,15,31,63,
     62
161Ø DATA 126,252,192,192,
```



```
     128,0,0,0,3,7,0,0,0,0,0,
     0,128,223
1620 DATA 127,63,31,31,15,7,
     7,3,0,128,128,192,192,
     192,224,240,252,126,1
     26,127
1630 DATA 63,63,31,31,15,15,
     7,3,3,131,131,131
1640 DATA 255,255,231,231,2
     55,252,172,183,128,19
     2,224,224,224,224,224
     ,224
1650 DATA 1,1,0,0,0,1,3,7,248
     ,255,255,255,255,255,
     255,255,0
1660 DATA 255,255,255,255,2
     55,255,255
1670 DATA 15,255,255,255,
     255,255,255,255,199,2
     55,255,255,255,255,25
     5,255
1680 DATA 223,239,255,255,2
     52,128,0,128,224,240,
     248,248,240,96,0,0,0,0
     ,200
1690 DATA 232,248,248,120,1
     12
1700 DATA 7,15,15,31,31,63,6
     3,255,255,255,255,255
     ,255,255,255,192,224,
     224
1710 DATA 224,224,224,192,1
     92,0,0,1,1,3,7,14,60,24
     0,224,192,128,128,0,0,
     0
1720 DATA 0,0,0,0,0,3,15,31,
     63,63,63,63,126,254,2
     52,240,191,191,63,63,
     63
1730 DATA 63,63,63,15,7,7,7,
     7,7,7,7,192,131,135,15
     9,191,255,254,252
1740 DATA 252,248,240,224,1
     92,128,0,0,0,1,3,7,6,6,
     4,0,127,254,240,224
1750 DATA 64,64,0,0,192,0,0,
     0,0,1,1,3,63,31,63,127,
     254,252,248,240
1760 DATA 7,3,0,0,0,0,0,0,24
     0,192,0,0,0,0,0,0,7,7,1
     5,31,31,31,23,23,
     224
1770 DATA 192,128,0,0,0,0,0,
     7,6,4,0,0,0,0,0
```

Todas estas adiciones se adaptan perfectamente a la primera parte del programa, y lo amplían para añadir al mismo más figuras. La línea 51 añade otro bucle FOR ... NEXT para introducir los datos adicionales de los nuevos caracteres. Las líneas 100 y 120 crean el mono, la línea 140 crea la serpiente y la línea 155 coloca una luna en el escenario. Para ahorrar memoria, los UDG del mono se asignan a la variable M$. Por carecer de comandos BASIC de alta resolución, la luna se representa con gráficos corrientes.



EL CUADRO COMPLETO

Como podrás ver cuando ejecutes el programa, la pantalla queda ahora llena de objetos y personajes, con lo que el cuadro parece completo. La serpiente, el mono y el sol son añadidos por las líneas adicionales.

También puedes añadir más elefantes, árboles o cocodrilos si lo deseas, pero en cualquier caso, dispones de un juego completo de personajes para utilizarlos en tus propios cuadros.

SIGUIENTES PASOS

La gama de variaciones que puedes obtener con los UDG es casi ilimitada. Además de cambiar el número de serpientes, de árboles, de monos, etc., también puedes utilizar los caracteres gráficos para que formen estas imágenes para otras cosas.

Un ejemplo evidente consiste en emplear las copas de los árboles con un color diferente (como puede ser el blanco) para representar nubes, o como arbustos.

En estos casos deberás tener en cuenta los contrastes de color.



Supón que quieres emplear copas de árboles como arbustos: si colocas los arbustos en las colinas, como éstas son verdes, dichos arbustos no se verán, a menos que desees que los arbustos sean rojos, por supuesto.

ANIMACION DE CUADRO

Las ventajas de crear imágenes con UDG no termina con las posibilidades de modificar la escena una vez terminada, sino que, con muy pocos cambios, puedes convertirla en un cuadro animado.

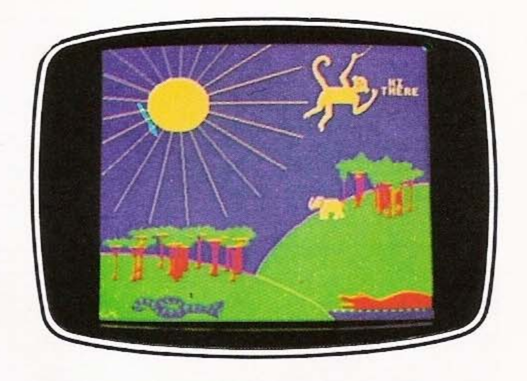
Por ejemplo, para animar los animales, puedes definir uno o más caracteres adicionales: otra trompa para el elefante, una serpiente con la cabeza levantada o el mono comiéndose un plátano son sólo algunas de las muchas posibilidades.

Una vez hayas decidido qué quieres que hagan los animales y hayas definido los UDG adicionales, te bastará con seguir el procedimiento para la animación descrito en los capítulos anteriores.

Empleando una serie de UDG distintos para las diferentes posiciones y presentándolos sucesivamente, el movimiento se podrá hacer mucho más real, más divertido, más amenazador, o como desees.

COMO SE ENCAJAN LOS UDG

Si vas a emplear la animación con UDG o a crear imágenes por ti mismo podrás encontrar en estas páginas los



dibujos de la serpiente y del mono y el del cocodrilo en el capítulo anterior.

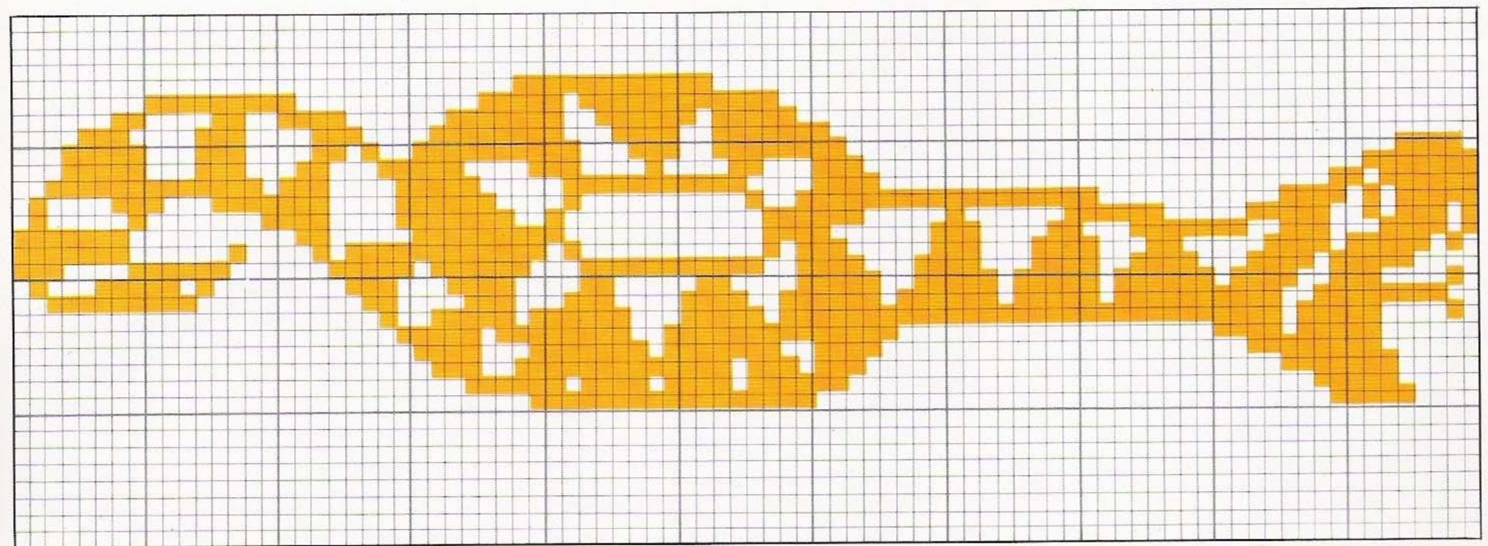
También se indican algunas posibles sustituciones de datos para el elefante, que es posible que desees emplearlas para la animación.

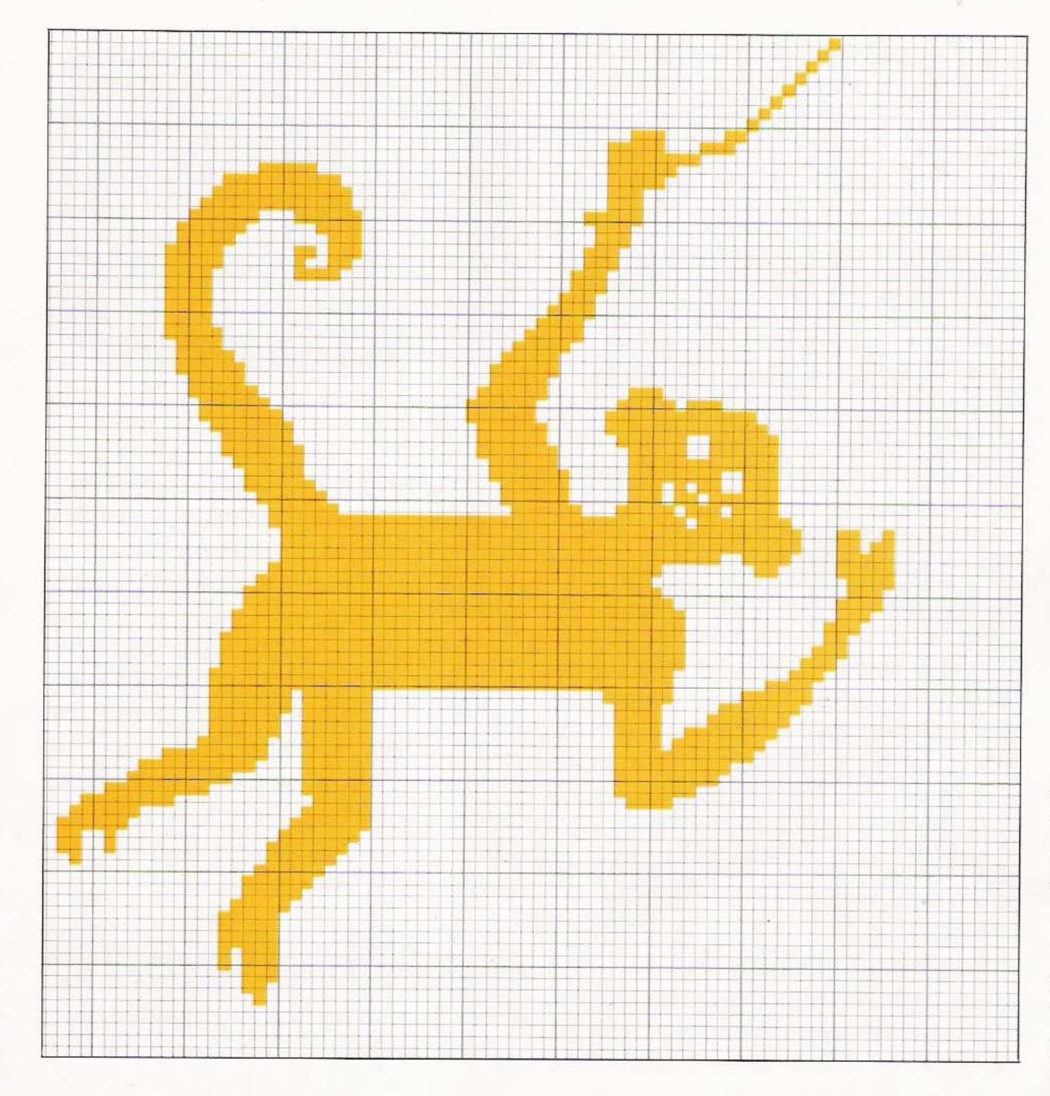
Mientras los animales parecen quedar limitados a escenas de jungla o de zoológico, los árboles, las nubes y los arbustos podrás utilizarlos en casi todos los cuadros que desees realizar con un programa. Con métodos similares podrás dibujas por ti mismo las escenas que desees.

AHORRANDO MEMORIA

El programa descrito emplea una gran cantidad de UDG para demostrarte la forma de utilizar tu ordenador más allá de sus límites, así como proporcionarte algunos personajes y figuras que puedes emplear en tus propios cuadros.

Pero si deseas o necesitas crear un cuadro impresionante con bastante menos UDG, puedes utilizar el mismo





UDG una y otra vez. Por ejemplo, puedes crear una manada entera de elefantes. O si en tu cuadro has incluido una pared, podrá cubrir una gran parte de la pantalla con sólo dos UDG.

Planificando cuidadosamente lo que deseas dibujar, podrás crear con mucho éxito cuadros muy interesantes con un número de UDG sorprendentemente reducido. Sin embargo, como es bastante normal que al cabo de un tiempo dispongas de un gran número de UDG, las únicas ventajas que pueden obtenerse «economizando» los UDG son el tiempo que ahorras al no tenerlos que teclear y en la memoria ocupada por los datos. Como el programa emplea una gran cantidad de memoria para almacenar los datos, puedes ahorrar mucha de esta memoria almacenando los bytes que forman los UDG en forma de bloques de memoria y borrando después las sentencias de los datos. Normalmente, el ordenador almacena cada byte dos veces: una en las posiciones de memoria que llenas con los datos (POKE) y también con las propias sentencias de datos.

El ordenador almacena en código máquina, o bytes de memoria, en la cinta alterando cuatro punteros para hacer que el área de BASIC corresponda exactamente con el bloque de memoria que deseas salvar. Entonces almacena los datos de la manera normal.

Por ejemplo, para almacenar los datos desde los UDG del programa, escribe en primer lugar:

```
POKE 43,0: POKE 44,48: POKE
45,230
POKE 46,55
```

y pulsa RETURN. A continuación entra esto:

```
SAVE 'DATOS UDG',1,1
```

El ordenador almacenará un bloque de memoria que empieza en el punto en que se inician los datos de caracteres, y termina al final de los mismos.

El 1 que hay inmediatamente después del nombre del fichero indica al ordenador que debe salvar en cinta, mientras que el segundo 1 que se trata de código máquina, o datos.

El cuarto POKE cambia las direcciones a las que se apunta el inicio y el final del BASIC. En números anteriores se explicaron cómo funciona esto para el inicio del BASIC (para los dos POKE 43 y 44) y para los otros dos POKE, 45 y 46, para el final del BASIC. Para volver a cargar los datos, emplea esto:

```
LOAD 'nombre de fichero',1,1
```

CONSEJOS PARA ANIMAR TUS FIGURAS

A continuación se indican algunos datos que puedes incorporar al programa principal para animar el elefante. Simplemente proporciona una alternativa de la posición de la trompa del animal, por lo que puedes pasar de una a otra para simular la animación.

```
DATA Ø,Ø,Ø,Ø,Ø,1,3,3
DATA 6,15,15,31,29,25,26,24
DATA 24,24,12,12,6,Ø,Ø,Ø
```

Además, tendrás que variar los correspondientes bucles FOR ... NEXT para establecer y presentar los nuevos UDG.

También puedes mejorar el elefante animado desplazándolo a una posición más visible de la que aparece en el cuadro.

La animación tiende a realzar un personaje, por lo que es una buena práctica emplear las partes animadas de tu cuadro como la parte principal. De esta manera podrás distraer la atención del observador de las partes menos llenas del cuadro.

La animación de los demás personajes podrás realizarla de manera parecida.

CONOS, CURVAS Y SECCIONES

- CORTAR UN CONO
- EL CIRCULO Y LA ELIPSE
- UNA PARABOLA Y UNA HIPERBOLA
- ROTACION DE CURVAS
- APLICACIONES PRACTICAS

La sencilla figura del cono es una de las formas geométricas más fascinantes de las matemáticas, y proporciona una entera familia de curvas. Con estos programas podrás investigar sus propiedades.

Las curvas han fascinado a los matemáticos desde tiempos inmemoriales, y cuanto más sencillas y elegantes, más importantes se consideraban. Los antiguos matemáticos griegos eran muy aficionados a hacer las matemáticas tan sencillas como se pudiera, y por ello cuando se descubrió que se podía obtener toda una familia de curvas (conocidas como secciones cónicas) de un simple tajo practicado sobre un cono, les pareció que los conos debían de revestir un significado especial.

De hecho la belleza de estas curvas está en que no son puras abstracciones matemáticas, sino que tropezamos con ellas en la vida diaria, y nos proporcionan una precisa descripción de los fenómenos físicos.

Naturalmente, en la naturaleza se encuentran otras curvas sencillas que no son secciones de un cono. La forma de una cuerda o de una cadena que cuelga entre dos puntos es una de ellas. Se llama catenaria, y, aunque parece una parábola, ambas curvas tienen sutiles diferencias, de tal modo que son descritas mediante ecuaciones muy diferentes. Pero las secciones cónicas son importantes ya que sirven para señalar el modo como se mueven las cosas, y, por ende, se necesitan para cualquier programa realista.

Algunas curvas son también útiles como objetos sólidos tridimensionales. Los cortes de un cono son bidimensionales, pero pueden rotar sobre sus ejes y ofrecer formas tridimensionales. Así el círculo se convierte en una esfera, con infinidad de usos, y la parábola se transforma en un paraboloide, empleado en cosas tan diversas como los faros de un coche, el juego de espejos de un telescopio, hornos solares, etc.

Este artículo se divide en dos partes. La primera describe cada una de las curvas y cómo dibujarlas en la pantalla, mientras la segunda ilustra el uso de estas curvas en simulaciones como la trayectoria de un cubo unido a una escalera (una elipse), o de una persona que cruza el río a nado (una parábola).

Verás también cómo se dibujan formas espectaculares con el solo empleo de la hipérbola y de la elipse.



CORTAR EL CONO

Las cuatro curvas que se obtienen cortando transversalmente un cono (el círculo, la elipse, la parábola y la hipérbola) son muy distintas entre sí. El primero que las estudió con detenimiento fue el griego Apolonio, allá por el año 200 a.C.

El punto inicial es el que formarían dos líneas que se cruzan como formando una X. Si se las hace rotar sobre un eje de simetría (véanse las ilustraciones) se obtienen dos conos que pueden cortarse de cuatro maneras diferentes.

Si se practica un corte en ángulo recto con el eje de simetría, la sección que se obtiene es un círculo.

El corte realizado formando un ángulo entre 90º y la mitad del ángulo que forman las líneas iniciales (denominado ángulo semivertical del cono), proporciona una sección denominada elipse.

El corte realizado formando un ángulo con el eje igual al ángulo semivertical nos da una parábola.

Un corte con un ángulo menor que el semivertical da una sección con dos partes denominadas hipérbola. La hipérbola tiene dos partes porque el corte afecta a ambos conos, el superior y el inferior.

Observa estos dos casos especiales. El corte que incluye el eje, es decir, que corta los conos en dos mitades, de arriba abajo, proporciona dos líneas rectas (las que sirvieron para generar los conos). Se trata de un caso especial de la hipérbola. En segundo lugar, haciendo un corte formando un ángulo de 90º con el eje y por entre los dos conos, lo que se obtiene es un punto, que, en realidad, es un círculo con radio cero.

Las figuras que te adjuntamos te serán suficientes para comprender cómo se obtienen todas estas formas geométricas descritas. Si te animas, tú mismo puedes construirte tus conos y realizar los cortes en diferentes direcciones, con servirte tan sólo de una cuartilla de papel. Los dos conos son necesarios para obtener una parábola, ya que con un solo cono obtendrás sólo una parte del modelo.

DIBUJAR LAS CURVAS

Todas las curvas se generan mediante sencillas ecuaciones, algunas de las cuales tú quizás ya conozcas. Los mandatos de gráficos de alta resolución para tu Commodore se obtienen mediante el cartucho BASIC de Simon o la utilidad en código máquina que ya te ofrecimos hace varios números.

EL CIRCULO

La ecuación de un círculo está dada por:

$X = A*\cos\theta$

$Y = A*\sin\theta$

donde A es el radio, X, Y un punto de la circunferencia, y θ (la letra griega *zeta* o *theta*) el ángulo formado por una línea fija, que habitualmente es el eje X.

Este primer programa dibuja un círculo de radio A, y centro el de la pantalla:

```
10 HIRES Ø, 1:COLOUR 1,1
15 A=6Ø
2Ø C=ATN(1)/45
3Ø XX=16Ø+A:YY=1ØØ
4Ø FOR TH=Ø TO 36Ø STEP1Ø
5Ø X=A*COS(TH*C):Y=A*SIN
   (TH*C)
6Ø LINE XX, YY,16Ø+X,1ØØ+Y,1
65 XX=16Ø+X:YY=1ØØ+Y
7Ø NEXT TH
8Ø GOTO 8Ø
```

LA ELIPSE

La ecuación de una elipse es muy similar a la del círculo. Para una elipse con el eje mayor igual a 2A y el eje menor igual a 2B, la posición de cualquier punto está dada por:

$X = A*\cos\theta$

$Y = B*\sin\theta$

La forma más o menos achatada de la elipse se determina por A y B. Cambia estas líneas:

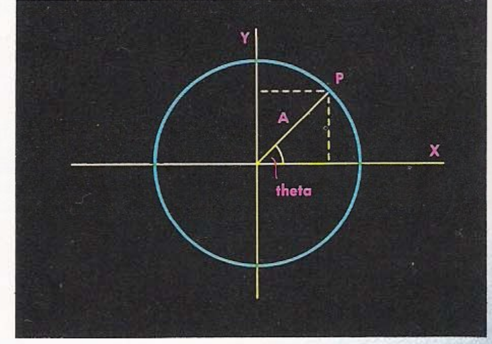
```
1Ø HIRES Ø,1:COLOUR 1,1
15 A=6Ø
16 B=3Ø
2Ø C=ATN(1)/45
3Ø XX=16Ø+A:YY=1ØØ
4Ø FOR TH=Ø TO 36Ø STEP1Ø
5Ø X=A*COS(TH*C):Y=B*SIN
   (TH*C)
6Ø LINE XX,YY,16Ø+X,1ØØ+
   Y,1
65 XX=16Ø+X:YY=1ØØ+Y
7Ø NEXT TH
8Ø GOTO 8Ø
```

LA PARABOLA

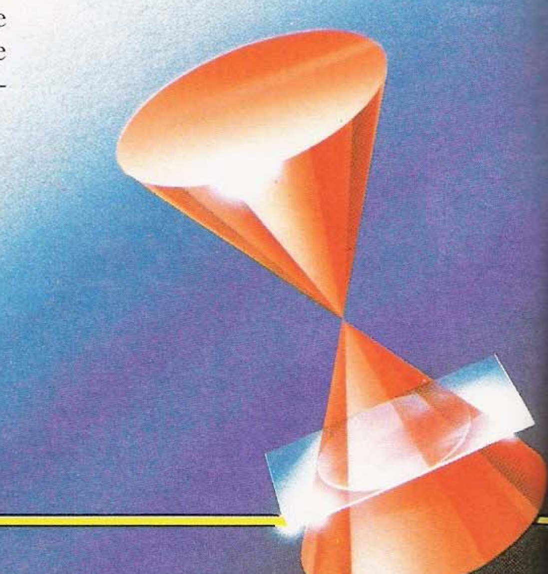
El tamaño de la parábola depende del valor de una variable T, y las ecuaciones son:

$X = T^2$

$Y = 2*T$

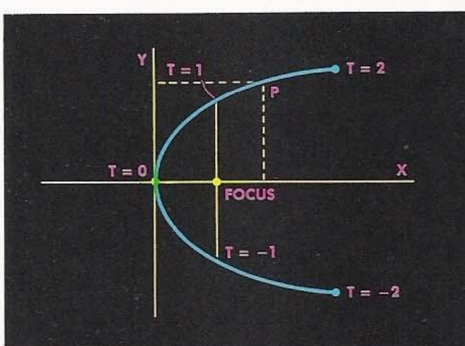


El círculo



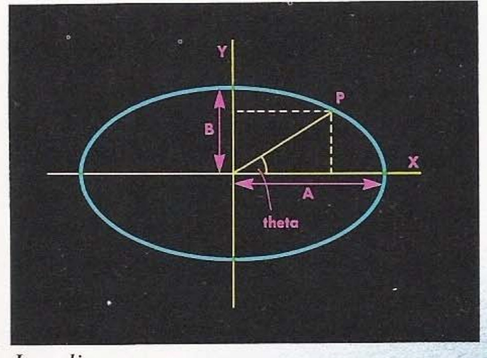
El valor T puede variar desde infinito hasta menos infinito, pero podemos hacer una sección de parábola bastante asequible entre T = 2 y T = –2. Estos valores deben ajustarse a escala en el programa mediante un factor M para que quepan en una pantalla de TV. Éste es el programa que dibuja la parábola:

```
1Ø  HIRES Ø,1:COLOUR
    1,1
15  M=23
2Ø  C=ATN(1)/45
3Ø  XX=16Ø+M*4:YY=1ØØ
    -4*M
4Ø  FOR T=-2 TO 2 STEP.Ø5
5Ø  X=M*T12:Y=2*M*T
6Ø  LINE XX,YY,16Ø+X,1ØØ+
    Y,1
65  XX=16Ø+X:YY=1ØØ+Y
7Ø  NEXT T
8Ø  GOTO 8Ø
```





La parábola



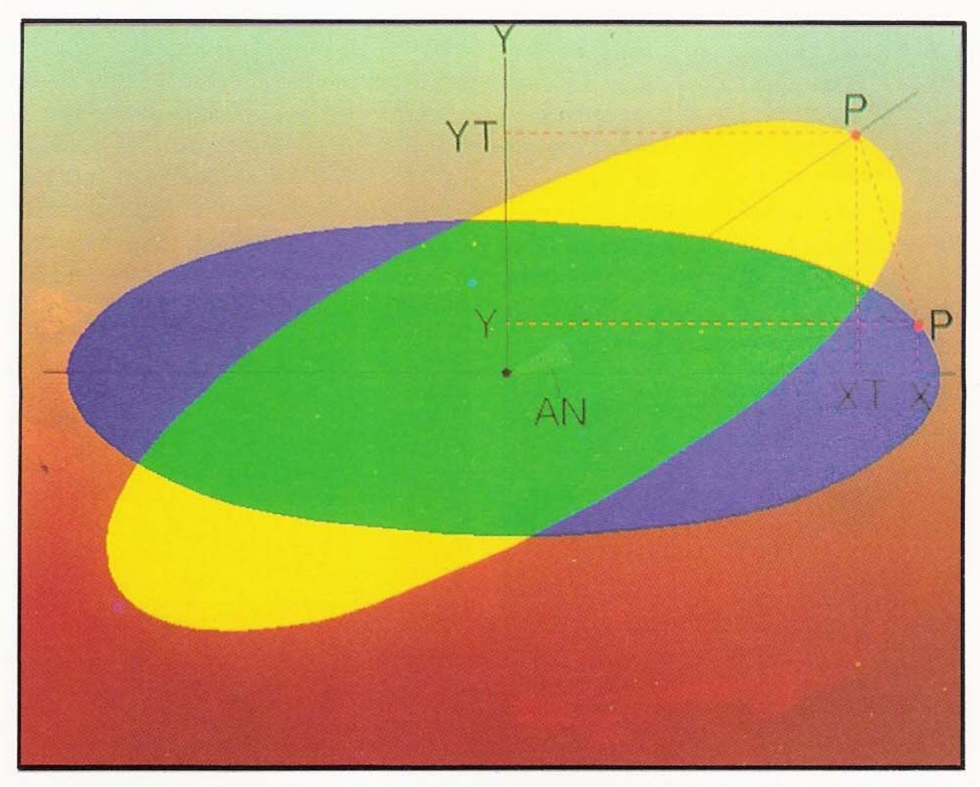
La elipse

LA HIPERBOLA

La hipérbola tiene por ecuación:
X = A/COS θ
Y = B*TAN θ

Una mitad de la hipérbola se ob-

La hipérbola



tiene cuando θ va de menos 90° a más 90°, y la otra mitad cuando va de más 90° a más 270°. Teóricamente es posible emplear sólo un bucle en el programa que lleve a θ desde –90° hasta +270°, pero tendríamos problemas en –90°, 90° y 270° donde se divide por cero, lo que el ordenador no sabe resolver. Incluso para valores de θ muy próximos a éstos, se obtienen valores enormemente grandes. El programa que presentamos emplea, por ello, dos bucles. Y nuevamente, mediante el factor M se puede dibujar a la escala deseada esta figura en la pantalla:

```
10 HIRES Ø,1:COLOUR 1,1
15 M=5Ø
2Ø C=ATN(1)/45
25 X=M/COS(–6Ø*C):Y=M*TAN
   (–6Ø*C)
3Ø XX=267:YY=8
4Ø FOR TH=–6Ø TO 6Ø STEP5
5Ø X=M/COS(TH*C):Y=M*TAN
   (TH*C)
6Ø LINE XX,YY,16Ø+X,1ØØ+
   Y,1
65 XX=16Ø+X:YY=1ØØ+Y
7Ø NEXT TH
75 X=M/COS(12Ø*C):Y=M*TAN
   (12Ø*C)
8Ø XX=5Ø:YY=8
9Ø FOR TH=12Ø TO 24ØSTEP5
1ØØ X=M/COS(TH*C):Y=M*TAN
    (TH*C)
11Ø LINE XX,YY,16Ø+X,1ØØ+Y,
    1
115 XX=16Ø+X:YY=1ØØ+Y
12Ø NEXT TH
13Ø GOTO 13Ø
```

HAGAMOS GIRAR LAS CURVAS

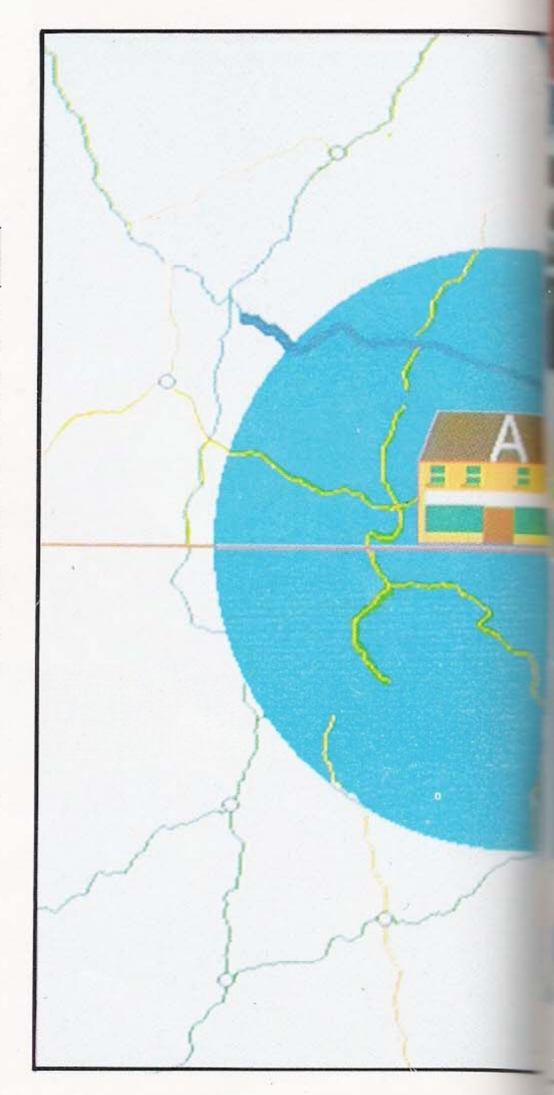
Los programas anteriores dibujaron las figuras de la manera más sencilla posible con X de eje horizontal e Y de eje vertical. Pero esto no siempre es lo que conviene, pues puede que necesites dibujar las curvas formando un determinado ángulo. La fig. 1 muestra lo que sucede a un punto del borde de una elipse cuando se le hace girar con un ángulo de AN grados. El punto P se mueve de la posición X, Y a su nueva posición XT, YT y sus nuevas coordenadas son:

XT = X*COS AN – Y*SIN AN
YT = X*SIN AN + Y*COS AN

He aquí la rutina de la rotación para tu ordenador:

```
1Ø HIRES Ø,1:COLOUR 1,1
15 A=6Ø
16 B=3Ø
17 AN=6Ø
2Ø C=ATN(1)/45
25 X=A:GOSUB 1ØØØ
3Ø XX=16Ø+XT:YY=1ØØ+YT
4Ø FOR TH=Ø TO 36Ø STEP1Ø
5Ø X=A*COS(TH*C):Y=B*SIN
   (TH*C)
55 GOSUB 1ØØØ
6Ø LINE XX,YY,16Ø+XT,
   1ØØ+YT,1
65 XX=16Ø+XT:YY=1ØØ+YT
7Ø NEXT TH
8Ø GOTO 8Ø
1ØØØ XT=X*COS(AN*C)–Y*SIN
     (AN*C)
1Ø1Ø YT=Y*COS(AN*C)+X*SIN
     (AN*C)
1Ø2Ø RETURN
```

En el programa que te dibuja la curva deberás introducir algunas modificaciones para que te sirva la rutina de la rotación. Se debe especificar (línea 17) el ángulo de rotación AN, la posición inicial ha de girar, y las líneas

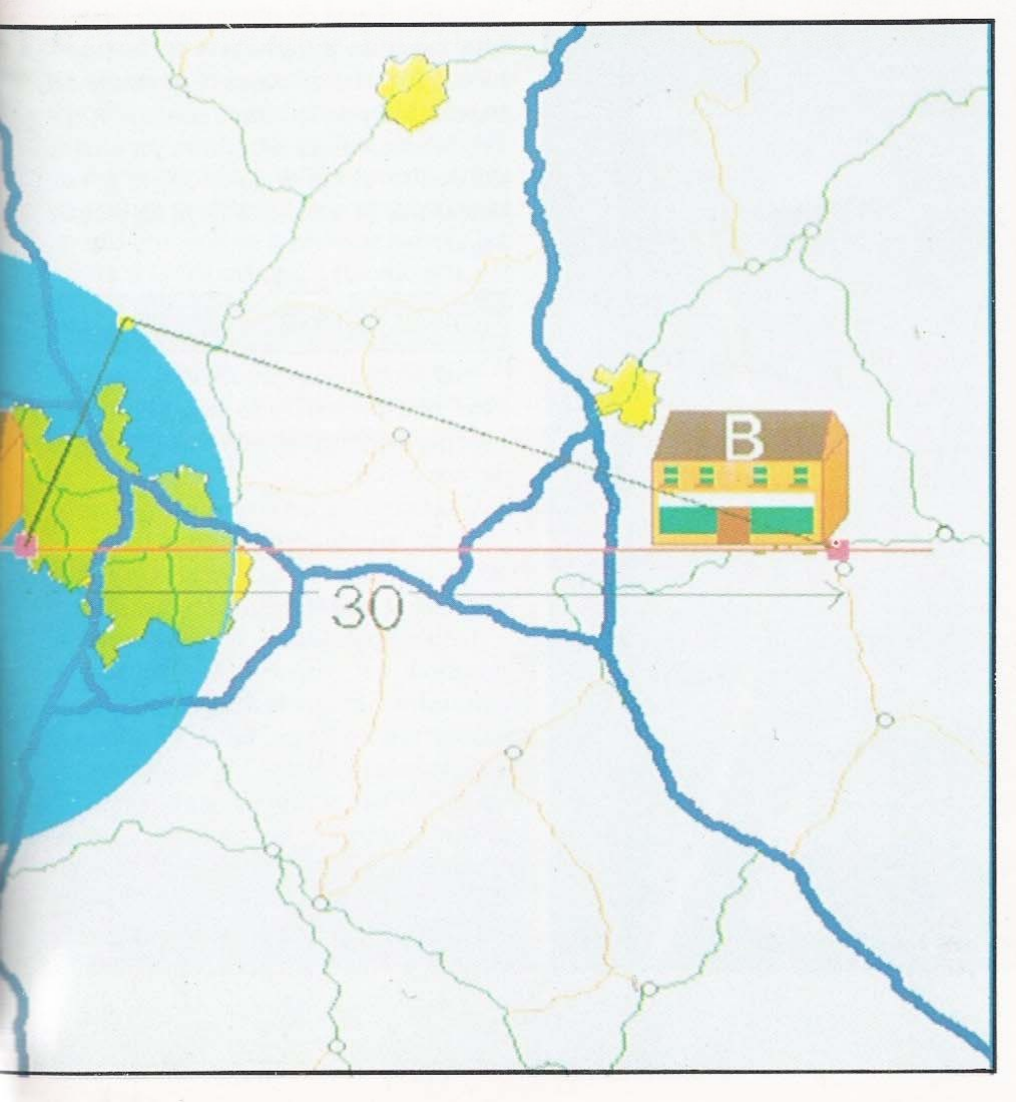


deben dibujarse según las nuevas coordenadas XT e YT en lugar de X e Y. Si lo deseas, puedes alterar la línea 17 para que te permita introducir (INPUT) el ángulo de giro.

He aquí los cambios que debes hacer en el programa que dibuja la elipse. No olvides añadir la rutina de la rotación:

```
10 HIRES Ø,1:COLOUR 1,1
15 M=23
17 AN=6Ø
2Ø C=ATN(1)/45
28 X=M*4:Y=–M*4:GOSUB
   1ØØØ
3Ø XX=16Ø+XT:YY=1ØØ+
   YT
4Ø FOR T=–2 TO 2 STEP.Ø5
5Ø X=M*T12:Y=2*M*T
55 GOSUB 1ØØØ
6Ø LINE XX,YY,16Ø+XT,
   1ØØ+YT,1
65 XX=16Ø+XT:YY=1ØØ+YT
7Ø NEXT T
8Ø GOTO 8Ø
1ØØØ XT=X*COS(AN*C)–Y*SIN
     (AN*C)
1Ø1Ø YT=Y*COS(AN*C)+X*SIN
     (AN*C)
1Ø2Ø RETURN
```

```
1Ø HIRES Ø,1:COLOUR 1,1
15 M=35
17 AN=6Ø
2Ø C=ATN(1)/45
25 X=M/COS(–6Ø*C):Y=M*TAN
   (–6Ø*C)
28 GOSUB 1ØØØ
3Ø XX=16Ø+XT:YY=1ØØ+YT
4Ø FOR TH=–6Ø TO 6Ø STEP5
5Ø X=M/COS(TH*C):Y=M*TAN
   (TH*C)
55 GOSUB 1ØØØ
6Ø LINE XX,YY,16Ø+XT,
   1ØØ+YT,1
65 XX=16Ø+XT:YY=1ØØ+YT
7Ø NEXT TH
75 X=M/COS(12Ø*C):Y=M*TAN
   (12Ø*C)
78 GOSUB 1ØØØ
8Ø XX=INT (16Ø+XT):YY=INT
   (1ØØ+YT)
9Ø FOR TH=12Ø TO 24Ø
   STEP5
1ØØ X=M/COS(TH*C):Y=M*TAN
    (TH*C)
1Ø5 GOSUB 1ØØØ
11Ø LINEXX,YY,16Ø+XT,
    1ØØ+YT,1
115 XX=16Ø+XT:YY=1ØØ+ YT
12Ø NEXT TH
13Ø GOTO 13Ø
1ØØØ XT=X*COS(AN*C)–Y*SIN
     (AN*C)
1Ø1Ø YT=Y*COS(AN*C)+X*SIN
     (AN*C)
1Ø2Ø RETURN
```



APLICACIONES PRACTICAS

Todas estas curvas pueden tener ciertas aplicaciones prácticas.

El círculo tiene tantos usos que no es posible enumerarlos. La rueda es un ejemplo demasiado trivial de un círculo, y un cojinete de bolas es también otro uso obvio de la esfera. Las esferas, o sus aproximaciones, se presentan con frecuencia en la naturaleza. Los ejemplos que se pueden poner van desde las gotas de lluvia y los granos de guisante hasta los planetas. Pero las esferas pocas veces son perfectas debido al efecto de la gravedad, el viento u otras fuerzas. Un planeta que gira en torno a una estrella podría describir una circunferencia, pero lo más frecuente es que describa una elipse.

Una aplicación práctica de gran utilidad del círculo es la que permite determinar el menor coste de transporte de una mercancía que puede comprarse en uno de dos puestos de distribución. Por ejemplo, supongamos que tú deseas comprar un ordenador que se vende tanto en la casa A como en la B y que distan entre sí 300 kilómetros. La casa A envía sus ordenadores a través de medios de transporte especiales y a un precio de 3 ptas. por km, mientras que la casa B emplea su propia furgoneta, que viene

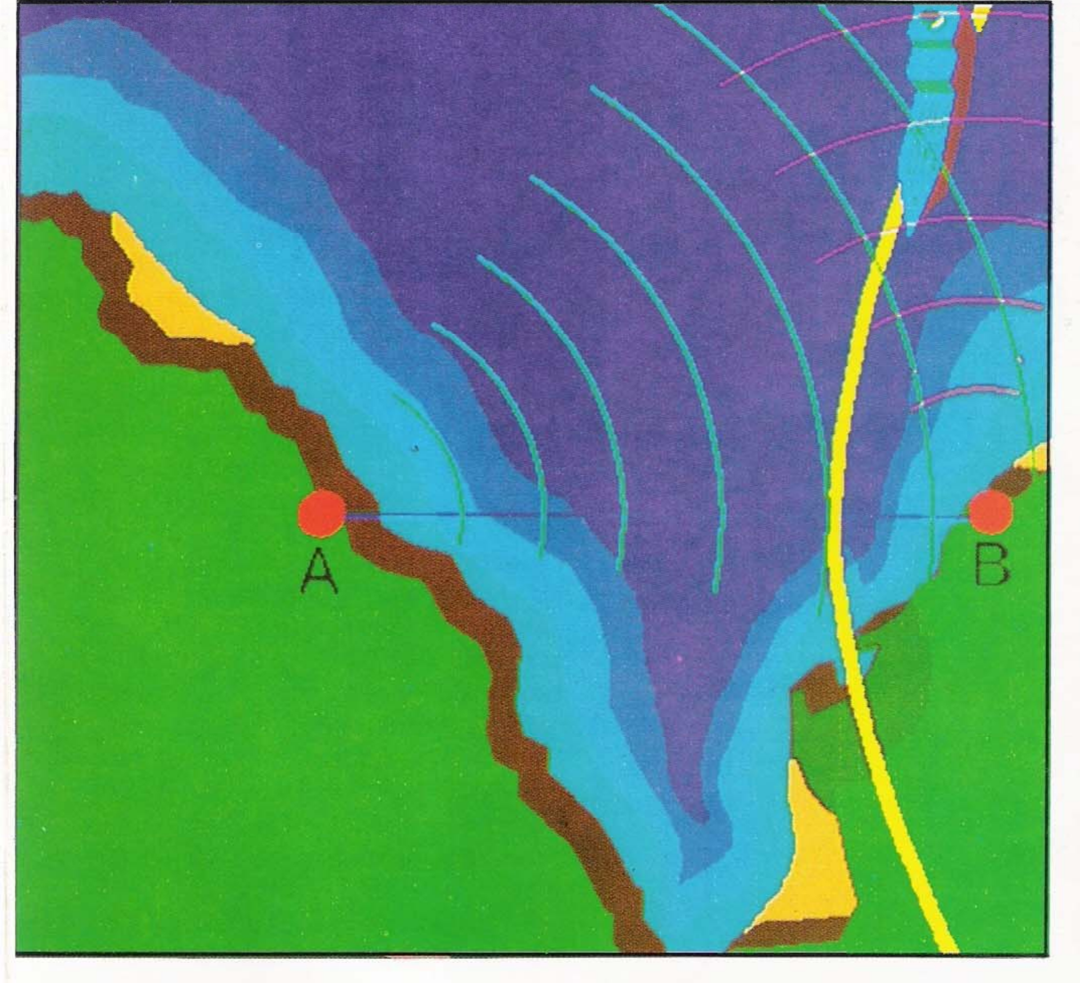
a salir a 1,5 ptas./km. Es inmediato marcar el área en el mapa de donde se puede obtener el ordenador deseado, al menor precio de transporte, sea de la casa A o de la B. La idea es marcar todos los puntos donde ambos costes son iguales, y unirlos por una línea. Lo que habrás hecho es marcar los puntos en que la distancia hasta B es el doble que la distancia hasta A.

Un punto estará en la línea entre A y B, a 100 km de A y 200 km de B (ya que 100 × 3 es igual que 200 × 1,5). Otro punto está en la misma línea a 300 km de A pero en la dirección opuesta a B (300 × 3 es igual que 600 × 1,5). Si tú unes todos estos puntos obtendrás un círculo de radio 200 km, conforme se muestra en la fig. 2. Si resides dentro del círculo es más barato comprarle a A, y si vives fuera de él es más barato comprarle a B.

La elipse tiene también sus usos prácticos. Si proyectas la sombra de una elipse sobre una superficie plana es posible mantener la elipse de tal manera inclinada que su sombra dé un círculo. Esta propiedad se utiliza en las válvulas de los canales circulares, donde se utiliza una aleta elíptica para controlar el gas o el fluido que circula. La aleta se ajusta a la tubería justo cuando alcanza el ángulo adecuado y de esta forma cierra la tubería.

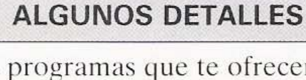
La parábola describe, como sabes, la trayectoria de un proyectil disparado al aire. Los cometas pueden viajar en forma parabólica alrededor del sol.

Una propiedad muy útil de la parábola es que los rayos de luz, calor u otro cuerpo paralelo al eje se reflejan a través del foco. Esta propiedad es válida en ambos sentidos, por lo que una bombilla eléctrica colocada en el foco dará un haz de luz paralelo, como el que se usa en los faros de los automóviles. En la otra dirección, los rayos paralelos que proceden del sol se pueden concentrar en el foco para obtener altísimas temperaturas como ocurre en los hornos solares.



En la práctica, los reflectores empleados para tales propósitos son paraboloides tridimensionales. Otro uso de los paraboloides se halla en los discos que sirven de antena de radar o de radio, donde la antena se coloca en el foco y puede emplearse tanto para transmitir como para recibir señales.

Una importante característica de la hipérbola consiste en el hecho de estar formada de dos partes. Y un uso práctico de ello está en el sistema de radar de los barcos. El sistema se basa en dos estaciones de radar. Una estación transmite señales normalmente, y la otra sólo se limita a retransmitir las señales recibidas por la primera estación. Cualquier barco que se halle en las proximidades recibe ambas señales y anota el tiempo que media entre sus llegadas. Si continúa moviéndose de tal modo que conserva esta diferencia de tiempo constante seguirá una trayectoria hiperbólica como la que mostramos en la fig. 3. Si el barco recibe también las señales de otras dos estaciones de radar y de nuevo toma nota de la diferencia de tiempo hará posible una segunda hipérbola y la intersección de ambas ofrecerá la posición del barco. No puede haber confusión sobre la rama de la hipérbola en que se encuentra el barco puesto que puede detectarse la señal que llega en primer lugar.



ALGUNOS DETALLES

Los programas que te ofrecemos en este artículo han sido pensados para una perfecta utilización de la pantalla de televisión.

Cuando los uses en tus propios programas, habrás de cambiar el factor M de incremento, para que las curvas se dibujen al tamaño que deseas.

Debes tener también cuidado con la parábola que gira o la hipérbola cerciorándote de que los cabos de las curvas entren en la pantalla. Para evitar que se salgan, altera los finales de los bucles FOR ... NEXT en la línea 40 del programa de la parábola y las líneas 40 y 90 del programa de la hipérbola. Para encontrar los límites exactos deberás recurrir al método de ensayo y error.

LEJOS DEL MUNDANAL RUIDO

- APUNTAR AL OBJETIVO
- DISTANCIAS
- LA VUELTA AL MUNDO
- MANIOBRAS EN EL ESPACIO
- MOVIMIENTOS PLANETARIOS

Uno de los sueños del hombre ha sido siempre alcanzar las estrellas. Este artículo, complemento del que dedicamos a los objetos voladores, va a enseñarle a tu micro cómo debe conseguir tal objetivo.

En el artículo en que tratamos de trayectorias, publicado en INPUT COMMODORE EXTRA VERANO 86, pudiste ver cómo se puede dividir la velocidad de un proyectil en un componente vertical y otro horizontal. Pudiste ver también cómo se puede modificar el alcance de un proyectil cambiando el ángulo de elevación y la velocidad a la que se disparaba. Este artículo reanuda en este punto el estudio de los cuerpos móviles. Analizará el movimiento de los objetos bajo la ley de gravedad, y te va a permitir conocer su trayectoria desde puntos cercanos a la superficie terrestre hasta muy lejanos, hasta quedar en órbita.

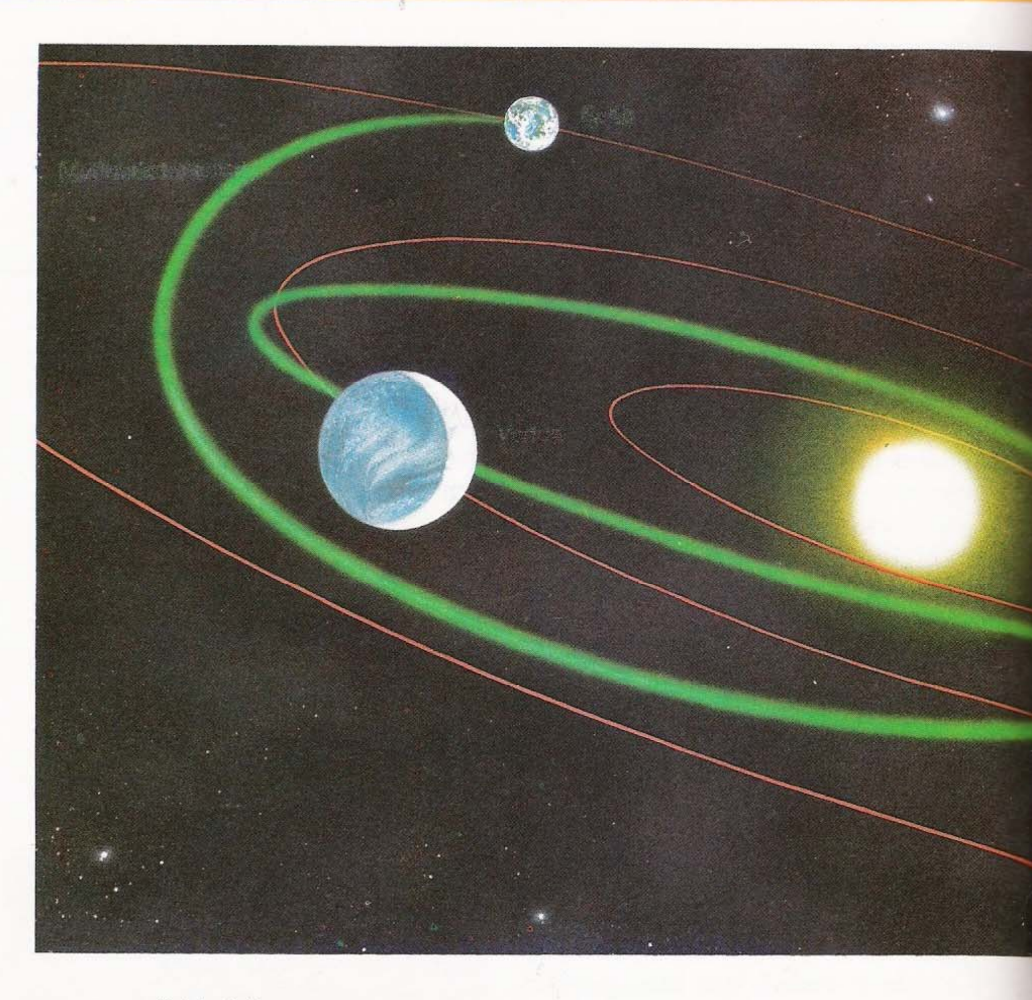
Antes de lanzarte al espacio, entra el primer programa, que te demuestra lo útil que nos va a resultar saber algo sobre trayectorias para cuando quieras hacer que tus juegos de proyectiles sean más interesantes y difíciles. Para tu Commodore 64 necesitas ajustar el modo gráficos en alta resolución, en todos los programas excepto el primero. Puedes servirte del cartucho de BASIC de Simon o bien entrar primero la utilidad de alta resolución en código máquina que te ofrecimos en esta revista. Si usas esta última deberás poner un prefijo a todos los mandatos de alta resolución.





```
99  GOSUB 13ØØØ
1ØØ  A%=RND(1)*8:
     B%=RND(1)*8+31: C%=R
     ND(1)*8+2:CT=Ø
11Ø  CT=CT+1
115  PRINT
     '[SHIFT+CLR/HOME]':SYS
     832
12Ø  GOSUB 9ØØØ
13Ø  GOSUB 7ØØØ
14Ø  GET I$:IF I$<>CHR$(13)
     THEN 14Ø
15Ø  GOSUB 95ØØ
16Ø  PRINT '[SHIFT+CLR/HOME]'
17Ø  INPUT '[CLR/HOME][CRSR
     ABAJO]ANGULO DE  DISPA
     RO'; A2
18Ø  IF A2<Ø OR A2>=9Ø
     THEN17Ø
19Ø  INPUT '[CLR/HOME][
     3*CRSR ABAJO]VELOCIDAD
     '; E
2ØØ  IF E=Ø THEN19Ø
2Ø5  GOSUB 9ØØØ
21Ø  AN=A2*(EXEC /18Ø):X3=8
22Ø  X=X3+8*(A%+1)
23Ø  H=8+X3*TAN(AN)-X3 ↑ 2/(
     E ↑ 2*COS(AN ↑ 2)
24Ø  Y=191-H:GOSUB
     1ØØØØ:FR=H/2+2Ø:WF=3
     3:GOSUB  11ØØØ
25Ø  X3=X3+7
26Ø  BY=24576+((Y-1) AND
     248)*4Ø+(X AND 5Ø4)+((Y-
     1) AND 7)
265  PO=(PEEK(BY) AND 2 ↑ (7-X
     AND 7))
27Ø  IF X<311 AND H>=8 AND
     H<183 AND PO=Ø
     THEN22Ø
273  HIT=ABS(X-(B%*8+4))<6
     AND H<24
275  IF PO>Ø OR HIT
     THENFR=2Ø: WF=129:
     GOSUB 11ØØØ
28Ø  FOR D=1 TO 2ØØØ:NEXT
29Ø  GOSUB 95ØØ
31Ø  IF CT<11 AND NOT HIT
     THEN11Ø
32Ø  PRINT '[SHIFT+CLR/HOME]'
```

```
322 FOR D=1 TO 99:NEXT
325 GOSUB 95ØØ
33Ø IF HIT THENPRINT 'BUEN
    DISPARO !!!':PRINT :PRINT
    'LO CONSEGUISTE EN'CT
34Ø IF NOT HIT THEN PRINT
    'FALLASTE !!!'
35Ø PRINT: PRINT 'INTENTALO
    DE NUEVO ...'
36Ø FOR D=1 TO 4ØØØ:NEXT
37Ø GOTO 1ØØ
7ØØØ V=1:FOR I=31936+A%*8
     TO 31936+A%*8+7:
     POKE I,V:V=V*2:NEXT
7Ø1Ø FOR I=31936+B%*8 TO
       31936+B%*8+7:POKE
       I,255:NEXT
7Ø2Ø FOR X=1 TO C%:FOR Y=1
     TO C%:BA=32256+(13+
     X)*8-Y*32Ø
7Ø3Ø FOR I=BA TO BA+7:POKE
     I,63:NEXT
7Ø4Ø NEXT :NEXT
7Ø99 RETURN
9ØØØ POKE 56576,15Ø:POKE 5
     3265,187:POKE 53272,
     29:RETURN
95ØØ POKE 56576,151:POKE
     53265,27:POKE 53272,
     21:RETURN
1ØØØØ BY=24576+(Y AND
      248)*4Ø+(X AND 5Ø4)+(
      Y AND 7):POKE BY,PE
      EK(BY) OR 2↑(7-(X AND
      7))
1ØØ1Ø RETURN
11ØØØ POKE 54296,1Ø
11Ø1Ø POKE 54278,251
11Ø2Ø POKE 54276,WF
11Ø3Ø POKE 54273,FR
11Ø35 FOR D=1 TO 2Ø:NEXT
11Ø4Ø POKE 54276,WF-1
11Ø5Ø RETURN
12ØØØ DATA 169,Ø,133,251,
      169,96,133,252,169,Ø,
      168,145,251,2ØØ,2Ø8,
      251
12Ø1Ø DATA 23Ø,252,165,252,
      2Ø1,128,2Ø8,24Ø
12Ø2Ø DATA 162,Ø,169,7,157,
      Ø,68,157,Ø,69,157,Ø,7
      Ø,157,232,7Ø,232,2Ø8
      ,241,96
13ØØØ FOR Z=832 TO
      875:READ X:POKE Z,
      X:NEXT Z:RETURN
```



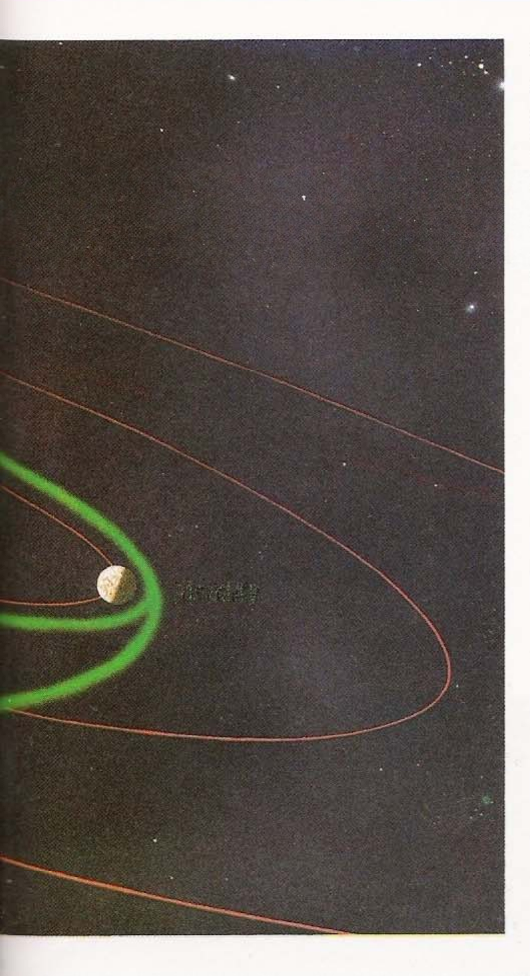
El programa pide que entres la velocidad de despegue y el ángulo de elevación para obtener un disparo que va desde la parte inferior izquierda de la pantalla hasta un objetivo que se halle en la parte inferior derecha. Puedes hacer más difícil el juego colocando el punto de disparo y el objetivo a distancias aleatorias (*random*) para cada serie de intentos, y, todavía más difícil, instalando una barrera de tamaño aleatorio en un punto también aleatorio entre los puntos de partida y de llegada. Cualquiera de las trayectorias que escojas habrá de tener la suficiente altura como para superar la barrera al efectuarse el disparo.

DISTANCIAS

Este programa es un buen ejemplo de cómo el cerebro humano emite sus juicios. Con sólo mirar las posiciones del fusil, del obstáculo y del objetivo, se puede estimar la velocidad y el ángulo requeridos para obtener una curva que hace que el proyectil se eleve por encima del obstáculo y dé en toda la diana. Con un poco de práctica, vas a sorprender por la cantidad de proyectiles que consigues que den en el objetivo.

Pero medir tales distancias de momento te resultará un tanteo con muchos fallos, por carecer de una vista lateral clara. Lo que puede ocurrir es que sepas la distancia aproximada del objetivo, y tengas que calcular el ángulo y la velocidad adecuadas. Analizando si el disparo te ha quedado corto o demasiado largo, progresivamente harás cálculos más ajustados.

El siguiente programa muestra cómo hacer esto. Despeja de la memoria el primer programa (CLEAR), pero una vez que lo has guardado (SAVE) si es que deseas volverlo a usar. Entra ahora estas líneas que siguen. Ajusta primero, no lo olvides, el cartucho de BASIC de Simon o la rutina en código máquina.



```
20 PRINT '[SHIFT+CLR
   /HOME]VELOCIDAD DE
   DISPARO (1-10000
   M/S)':INPUT SP
30 IF SP<1 OR SP>10000
   THEN20
40 PRINT
   '[SHIFT+CLR/HOME]ANGULO
   DE DISPARO (1-90
   DEG)':INPUT A
50 IF A<1 OR A>89 THEN40
60 A=A*(EXEC /180)
70 R=SP*SP*SIN(2*A)/10
80 PRINT
   '[SHIFT+/CLR/HOME]EL
   ALCANCE ES',INT (R+.5);'M
   ETROS'
90 PRINT '[CRSR ABAJO]PULSA
   UNA TECLA PARA SEGUIR (-
   0- PARA SALIR)'
100 GET D$:IF D$='' THEN100
110 IF D$<>'0' THEN RUN
```

Este programa te permite introducir (INPUT) los valores de la velocidad inicial (línea 20) y del ángulo de elevación (línea 40). La línea 70 calcula seguidamente y visualiza el alcance que darán al proyectil estos valores. La variable R es el alcance (*Range*), SP la velocidad (*SPeed*), A es el ángulo, y el 10 es el valor aproximado de la gravedad próximos a la superficie terrestre (en realidad es 9,81).

El efecto de la presión del aire no se tiene en cuenta, aunque no debes olvidarla en pruebas que simulen la realidad. La velocidad de disparo más alta que posee un cañón supera los 2.000 m/s. Con ella se han disparado objetos fuera de la Tierra en misión investigadora. Los objetos alcanzan una altura de 180 km, pero sin la presión del aire la altura podría llegar a ser de unos 250 km.

LA VUELTA AL MUNDO

Si el objeto es disparado con un ángulo que lo eleve por encima de la Tierra, el efecto de la presión es menor, pero si deseas un alcance aún mayor, habrás de tener en cuenta la curvatura de nuestro planeta. Uno de los primeros científicos que se plantearon el problema del máximo alcance de un proyectil que se eleva sobre la superficie de la Tierra fue el británico sir Isaac Newton.

Newton imaginó un poderoso fusil en la cima de una montaña lo suficientemente alta como para estar fuera de la atmósfera terrestre. Los disparos hechos a una velocidad cada vez mayor llegarían cada vez más lejos si la Tierra fuera plana. Pero dado que ésta es redonda, la superficie se separa del proyectil, con lo que puede alcanzar una distancia mayor que si nuestro planeta fuera plano.

Newton argumentaba además que se podría entonces disparar un proyectil a tanta velocidad que nunca cayera en tierra, sino que le alcanzara a él por la espalda. Cuando el proyectil fuera a caer sobre la superficie de la Tierra, ésta se curvaría de tal manera que el proyectil quedaría «sin tierra» para siempre: estaría dando vueltas al mundo.

Una vez que cualquier objeto escapa de la gravedad del planeta, su velocidad y distancia respecto del planeta determinan luego el tipo de órbita que seguirá: circular o elíptica.

Para que se puedan hacer predicciones y medidas sobre un planeta o satélite, debe conocerse su órbita con precisión: hay que saber las características exactas de la elipse. El grado de «achatamiento» de una elipse se denomina excentricidad (E). Ésta mide la proporción entre la largura y la anchura de la elipse. Si E es igual a 1, la elipse es igual de ancha que de larga, o sea, es una circunferencia.

Entra y ejecuta (RUN) el siguiente programa para ver el efecto obtenido al variar E con valores que van de cero y pico a mayores que uno:

```
30 PRINT '[SHIFT+CLR/HOME]E
   XCENTRICIDAD (0.1-1.9)':I
   NPUT E
40 IF E<.1 OR E>1.9 THEN30
50 HIRES 0,1:X=160:Y=100-
   E*50
60 FOR A=0 TO 2*EXEC +.2
   STEP.1
70 LINE X,Y,160+*SIN(A),100-(
   E*50*COS(A)),1
80 X=160+80*SIN(A):
   Y=100-(E*50*COS(A)):NEX
   T A
90 FOR Z=1 TO 2000:NEXT
   Z:NRM :RUN
```

Este programa establece un bucle entre las líneas 30 y 110 para dibujar elipses para las cuales tú introduces (INPUT) el valor de E en la línea 30. Las curvas se dibujan mediante un FOR ... NEXT (líneas 60 a 80) para que dibujen DRAW entre cada punto.

Entra valores de E dentro del intervalo que te muestra la pantalla y comprueba cómo E = 1 da una circunferencia, E = 0, ... te da elipses aplanadas, y E > 1 te proporciona elipses como huevos de pie. Notarás, sin embargo, que en tu Commodore, E = 1,5 es muy parecida a una circunferencia. Conclusión: toda órbita no es más que una elipse con su preciso valor de E.

MANIOBRAS EN EL ESPACIO

Una vez en órbita, un satélite o nave espacial no necesita energía alguna para mantenerse en ella, dado

que su caída es libre. Cualquier uso de cohetes lo desplazará a una órbita superior o inferior.

Cuanto menor sea el radio de la órbita, más rápidamente se moverá el objeto. Entra el siguiente programa para comprobarlo.

```
10 HIRES 0,1
20 POKE 54296,15:POKE
   54277,64
40 R=30:RT=INT
   (RND(1)*70)+10:IF ABS(R-R
   T)<20 THEN40
50 TEXT 157,97,'*',1,1,1
60 S=.1:A=0:AT=INT
   (RND(1)*10)+1:F=0
80 A=A+S
95 LT=AT
100 AT=AT+.1*SQR((40/RT)↑3)
110 GET A$:IF A$='[CRSR
    DCHA.]' AND R<95 THENF=
    F+1:R=R+1:S=S*SQR(((R
    -1)/R)↑3)
120 IF A$='[CRSR ABAJO]' AND
    R>8 THENF=F+1:R=R-1:S
    =S*SQR(((R+1)/R)↑3)
130 X=R*SIN(A):Y=R*COS(A)
140 XT=RT*SIN(AT):YT=RT* COS
    (AT)
150 PLOT 160+X,100-Y,1
160 POKE 54276,17:POKE
    54273,1+R:FOR Z=1 TO
    5:NEXT Z
165 POKE 54276,0:POKE
    54273,0
170 PLOT 160+XT,100-YT,1
175 PLOT 160+RT*SIN(LT),100-
    RT*COS(LT),0
180 IF ABS(X-XT)>3 OR ABS(Y-
    YT)>3 THEN80
190 NRM :PRINT
    '[SHIFT+CLR/HOME]HAS
    USADO';F;'ENCENDIDOS'
```

Ejecútalo (RUN) y observarás la estela de un aparato en órbita y también el satélite objetivo (sin estela). Intenta ajustar la órbita de tu aparato con la del satélite mediante las teclas de flechas de cursor arriba y cursor abajo. La línea 40 establece el radio R de la órbita del aparato a valor 200 y el radio del objetivo a un valor *random*. La línea 60 establece luego las variables para las posiciones de partida.

El punto crucial del programa está en la línea 100, que se inspira en otra importante ley física: la raíz cuadrada del tiempo que se tarda en hacer una órbita entera, dividida por el cubo del radio es constante. Es lo que explica SQR y la potencia a 3 de esta línea.

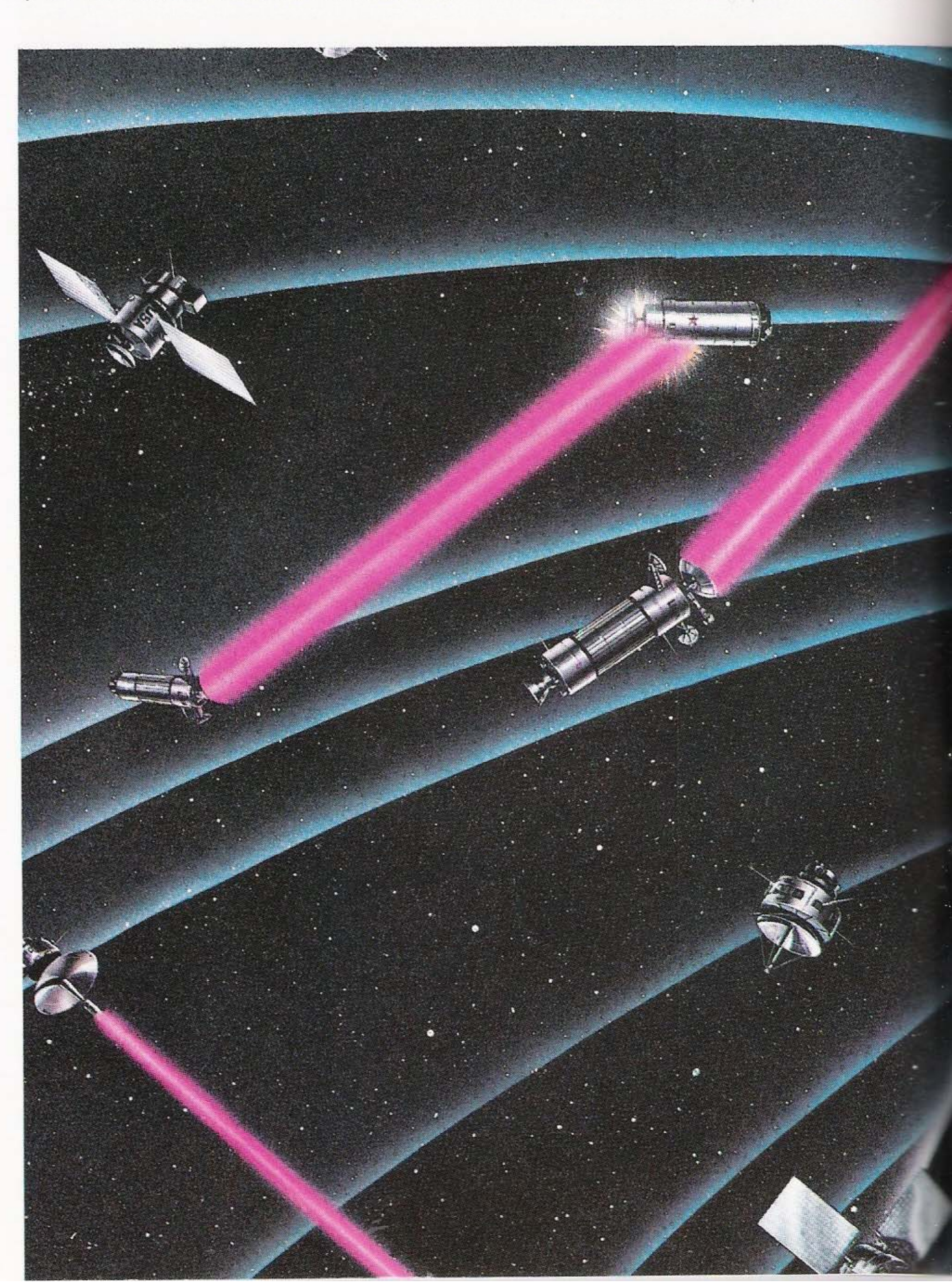
Para aumentar o disminuir el tamaño de la órbita (hacer maniobras) deberás utilizar las flechas de cursor arriba y abajo. Recuerda que cuanto más pequeñas son las órbitas a más velocidad irás.

MOVIMIENTO PLANETARIO

En realidad, las maniobras de salida y entrada a una órbita son mucho más laboriosas que lo que el programa anterior pueda hacer pensar. Es muy fácil trasladarse de una órbita elíptica a otra, pero la inmensidad del espacio hace que sea difícil localizar un objetivo. Y para complicar las cosas interviene de nuevo el efecto de la gravedad. La gravedad del Sol, de la Luna y de los planetas ejerce su influencia sobre la trayectoria del aparato espacial.

En la práctica las herramientas de los astronautas son precisamente unos poderosos ordenadores. Éstos se encargan de controlar la velocidad, el tiempo y la dirección del encendido de los cohetes para mantener la nave en la trayectoria deseada.

Una vez situada en la trayectoria co-





rrecta, la nave queda sometida al influjo del campo gravitatorio del sistema solar. Sólo son necesarias unas pocas correcciones en meses y años de viaje: justo cuando los planetas y el Sol se interponen año tras año. El siguiente programa va a permitirte observar los planetas en movimiento:

```
1Ø PRINT
   '[SHIFT+CLR/HOME][CTRL+1]'
   :NRM::COLOUR Ø,1
2Ø DIM D(8),P(8),I(8),A(8),
   B(8)
3Ø FOR T=Ø TO 8:READ D(T),
   P(T):NEXT T
4Ø PRINT
   '[SHIFT+CLR/HOME]CUANTOS
   PLANETAS (1-9)':INPUT
   S
5Ø IF S<1 OR S.>9 THEN
   4Ø
6Ø S=S-1:SC=D(S)/9Ø:T=P(S)/
   75
7Ø PRINT
   '[SHIFT+CLR/HOME]EXISTEN
   ';INT (T);'DIAS':PRINT 'DE
   INTERVALO ENTRE CADA
   PUNTO'
75 PRINT '[CRSR ABAJO]PULSA
   -ESPACIO- PARA
   SEGUIR'
76 GET A$:IF A$<>' ' THEN
   76
8Ø HIRES 1,Ø
11Ø G=EXEC /18Ø
12Ø FOR Q=Ø TO S
13Ø R=D(Q)/SC:A=R:B=R:E=Ø:
    P=P(Q)/T
14Ø IF Q=Ø
    THENE=.2:B=A*.98
15Ø IF Q=8
    THENE=.26:B=A*.96
155 IF S>3 AND Q<4 THEN
    245
16Ø IF P>3 THENP=INT (P+.5)
17Ø I(Q)=I(Q)+36Ø/P
18Ø Y=G*I(Q):X=INT
    (A*(COS(Y)-E)):Y=INT (B*S
    IN(Y))
2ØØ TEXT 157+A(Q),97-B(Q),
    '[SHIFT+W]',Ø,1,1
22Ø PLOT 16Ø+A(Q),1ØØ-B(Q),
    1
225 TEXT 157,97,'[SHIFT+Q],1,
    1,1
23Ø TEXT 157+X,97-Y,
    '[SHIFT+W]',1,1,1
24Ø A(Q)=X:B(Q)=Y
245 NEXT Q
25Ø M=M+T
26Ø GET A$:IF A$=' ' THEN
    RUN
27Ø GOTO 12Ø
28Ø DATA 58,88,1Ø8,225,15Ø,
    365,228,687
29Ø DATA 778,4333,1427,
    1Ø759,287Ø,
    3Ø685
3ØØ DATA 4497,6Ø19Ø,5969,
    9Ø741
```

Cuando ejecutes (RUN) este programa, deberás entrar un valor para seleccionar el número de planetas que deseas observar. Cuanto mayor sea el número (entre el 1 y el 9), más complicadas imágenes obtendrás. Para saber qué planetas estás observando, recuerda que el más próximo al Sol es Mercurio, después le sigue Venus, y luego la Tierra, Marte, Júpiter, Saturno, Urano, Neptuno y Plutón. Ejecuta el programa con valores diferentes y observa que las órbitas de dos planetas (Mercurio y Plutón) son claramente elípticas. De hecho todas son elípticas, sólo que algunas tienen una excentricidad muy pequeña y casi se asimilan a circunferencias.

RECURSION: BUCLES DENTRO DE BUCLES

- UN AIRE DE MISTERIO
- MAQUINA INTELIGENTE
- PROCEDIMIENTOS RECURSIVOS
- TRANSFERENCIA DE PARAMETROS
- DETECCION DE ERRORES

Si alguno de tus programas posee subrutinas a las que se llama repetidamente, puedes encontrarte con un caso adecuado para utilizar programación recursiva. Averigua cómo.

Esencialmente, programar un ordenador es un ejercicio de resolución de problemas. A medida que tu habilidad se desarrolle, podrás darte cuenta de que la mayor parte de problemas pueden resolverse si se dividen en problemas menores y más simples. Sin embargo, eventualmente llegarás a una de las pesadillas de los programadores, en la que un intento de resolver un problema te lleva solamente a un nuevo problema, la solución del cual te conduce a otro problema, y así sucesivamente.

Puedes considerar que se trata de un punto lo bastante delicado como para hacer un alto e ir en busca de un problema distinto, pero recuerda que tu micro no se ve afectado por esa clase de barrera mental que hace que un cerebro humano aproveche la menor oportunidad para salir de la pesadilla. De hecho, hay una técnica de programación avanzada que sirve para resolver ciertos tipos de problemas que pueden reducirse a problemas dentro de problemas. Dicha técnica se denomina recursión.

UN AIRE DE MISTERIO

Generalmente, los programadores inexpertos ven la recursión como un tema altamente complicado y misterioso. Eso es a causa de que las técnicas de programación utilizadas pueden ser extremadamente difíciles de seguir a partir del listado. Algunas veces, ni siquiera un diagrama de flujo muestra claramente lo que sucede realmente. No obstante, el principio no es difícil de captar.

En términos matemáticos, «recursión» es la repetición de una operación determinada. Sin embargo, esta definición no es



estrictamente aplicable a la programación, en donde tiene un significado más preciso. En esencia, recursión es una llamada primaria a una subrutina o procedimiento con un conjunto de parámetros iniciales. A continuación, la subrutina o procedimiento se llama a sí mismo repetidamente, actualizando los parámetros en cada llamada, hasta que se ha llevado a cabo una determinada tarea.

MAQUINA INTELIGENTE

Para ayudarte a comprender el concepto básico de recursión, piensa en cómo un conductor puede viajar desde el lugar A al lugar B, a través de una gran ciudad con la que no está familiarizado. El conductor dispone de mapas con las calles de la ciudad, pero observa que viajar directamente de A a B es demasiado difícil. Por tanto, decide descomponer el problema en un cierto número de problemas similares (pero mucho más simples), cada uno de los cuales puede irse resolviendo sucesivamente. El modo más sencillo de hacerlo consiste en seleccionar un lugar (C), que se encuentre entre A y B, y decidir cómo viajar desde A hasta C. A continuación conduce su coche hasta C.

Una vez en C, el conductor mira si puede viajar directamente desde C hasta B. Si es posible, lo hace. En caso contrario, repite el proceso anterior eligiendo un nuevo lugar (D), entre C y B. Este proceso se repite hasta que el conductor llega a B, su punto de destino.

Este ejemplo simple demuestra adecuadamente los principios de la recursión tal como se aplican a la programación de ordenadores. La solución de un problema difícil se describe en términos de problemas más sencillos (o más cortos).

A medida que se alcanza cada subtarea (o nivel) de la recursión, resulta a menudo necesario almacenar la información relativa a la posición anterior, alcanzada en el nivel precedente, para volver más adelante a ella. A medida que se entra en cada nivel, se obtiene un conjunto de parámetros distintos y se efectúa una comprobación para determinar si se ha completado la tarea completa. Sin dicha comprobación, el proceso no terminaría nunca. Para ver el método utilizado, introduce y ejecuta el primer programa, que imprime los enteros positivos desde un valor de entrada (N) hasta uno.

```
2Ø PRINT '[SHIFT+CLR/HOME][
   CRSR ABAJO]>[CRSR DCH
   A.][CTRL+9]N.ENTEROS
   POSITIVOS DESDE N HASTA 1'
3Ø PRINT :PRINT '[CRSR
   ABAJO][ 6
   ESPACIOS]ENTRAR EL
```

```
   VALOR ENTERO QUE DESEAS
   DESCONTAR (1-22)'
35 INPUT N:N=INT (N):IF N<1
   OR N>22 THENEND
40 GOSUB 80
50 GOTO 30
80 IF N=0 THENRETURN
90 PRINT N',';
100 N=N-1:GOSUB 80
110 RETURN
```

El programa te permite introducir el valor del mayor entero a partir del cual deseas contar. Si introduces un valor menor de uno, se detiene el programa. Los valores superiores a 22 también detienen el programa porque el micro sólo puede recordar 23 saltos a una subrutina. La línea 40 llama la subrutina recursiva, cuya primera línea comprueba si se ha completado la tarea entera.

Dicha comprobación es crucial para terminar las llamadas recursivas. Se entra en el primer nivel de recursión con el valor de N que has especificado.

Dicho valor se imprime en la línea 90. Se entra en el segundo nivel en la línea 100, que reduce el valor de N en uno y llama la subrutina otra vez con el nuevo valor de N. El programa continúa el bucle entre las líneas 80 y 100, imprimiendo sucesivamente cada entero.

Cuando N se reduce a 0 (en la línea 100), el programa bifurca del modo habitual a la línea 80, en donde esta vez deberá obedecer la instrucción RETURN. Esto provoca un retorno desde la subrutina llamada en la línea 100. La siguiente instrucción se encuentra en la línea 110, que provoca un retorno desde la subrutina llamada en la línea 40. La instrucción siguiente (línea 50) ejecuta de nuevo el programa.

Observa que el programa termina con un valor de N = 0, según la línea 100, pero que la línea 90 no imprime nunca dicho valor. Para restaurar N al último valor impreso, puedes teclear N = N + 1 como línea 105. A continuación, N tomará el último valor impreso.

PROCEDIMIENTOS RECURSIVOS



```
10 DIM N(34),A(34)
30 PRINT '[SHIFT+CLR/HOME][
   CRSR ABAJO]>[CRSR DCH
   A.][CTRL+9][ 2 ESPACIOS
   ]CALCULODE[ 3 ESPACIOS
   ]':PRINT '[ 2*CRSR DCHA.
   ][CTRL+9][ 2 ESPACIOS]F
   ACTORIALES[ 2 ESPACIOS
   ][CRSR ABAJO]'
40 PRINT 'ENTRAR EL N.
   FACTORIAL DESEADO (1-33)'
45 INPUT NU
50 IF NU>33 OR NU<>INT
   (NU) OR NU<0 THENRUN
60 IF NU=0 THENPRINT
   '[SHIFT+CLR/HOME]':END
70 LE=:1(LE)=NU:AN=NU
80 GOSUB 150
90 PRINT AN'![CRSR
   DCHA.]=[CRSR DCHA.]';A(1)
   '[CRSR ABAJO]':GOTO 40
150 IF N(LE)=0 THENA(LE)=1:
    GOTO 180
160 LE=LE+1:N(LE)=N(LE-1)-
    1:GOSUB 150
170 LE=LE-1:A(LE)=A(LE+1)*
    N(LE)
180 RETURN
```



Ejecuta el programa e introduce un valor en respuesta al mensaje. El programa calcula e imprime el *factorial* del número que hayas introducido, es decir el producto de cada entero desde uno hasta el propio número inclusive. Por ejemplo, 5 factorial (que se escribe 5!) es 1 × 2 × 3 × 4 × 5 = 120. Los cálculos factoriales se requieren a menudo en ciertas aplicaciones estadísticas, por lo que sería útil un mé-

todo simple de generarlos. Pero el método mejor depende del lenguaje que utilices.

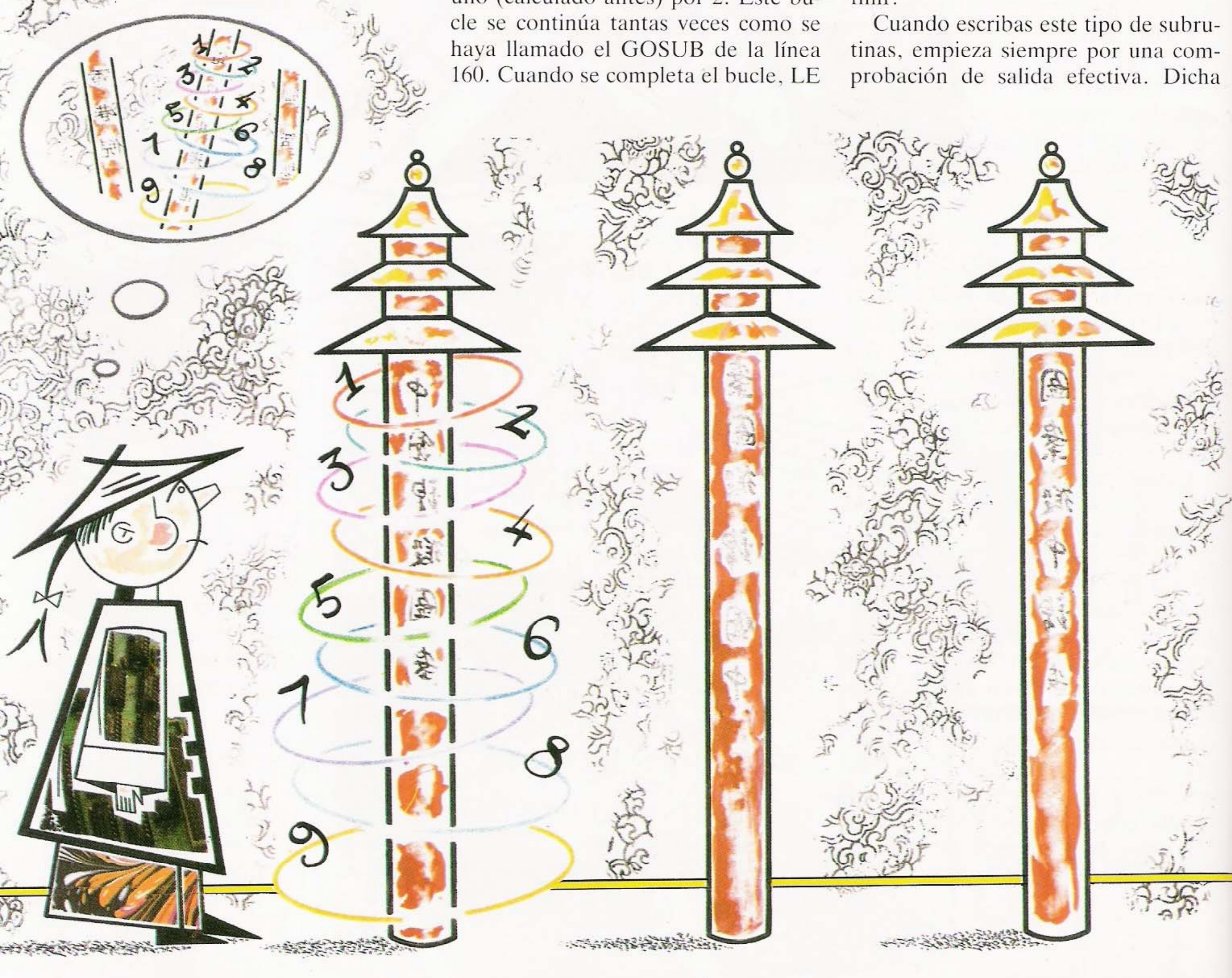
El programa dimensiona variables (línea 10) en número suficiente para completar la tarea. Dichas variables de matriz reservan espacio de memoria para su uso exclusivo, utilizando el nivel de recursión (LE) como subíndice de la variable que se esté utilizando en ese momento, N (LE).

Esencialmente, el programa empieza en la línea 70, en donde el nivel se pone en uno. Además, aquí se asigna el número que hayas introducido, por ejemplo cinco, a N (1) y a AN (la variable que acumula la respuesta). A continuación, la línea 80 llama al primer nivel de recursión. La primera línea de la rutina de recursión (línea 150) comprueba si se ha llegado al final del problema, cuando N (LE) = 0. Pero, por el momento, N (LE) es cinco, por lo que el control pasa a la línea 160. Esta incrementa el nivel (a dos), pone en 4 el número actual, luego llama de nuevo la rutina recursiva y así sucesivamente. Cuando la línea 160 incrementa el nivel a seis y decrementa el número actual a cero, la línea 150 detecta que N (6) es cero, por lo que el elemento seis de la matriz A se pone a uno y la línea 180 envía un RETURN al final de la línea 160. Ahora el control pasa a la línea 170, en donde el nivel se decrementa a cinco y A (5) toma un valor igual a A (6) veces N (5). Esto hace que A (5) sea igual a 1 × 1 = 1. Ahora la línea 180 devuelve de nuevo el control al final de la línea 150 en donde, esta vez, A (4) toma un valor igual a A (5) veces N (4). Por lo que A (4) es igual a uno (calculado antes) por 2. Este bucle se continúa tantas veces como se haya llamado el GOSUB de la línea 160. Cuando se completa el bucle, LE es igual a uno y el último RETURN va a la línea 80. La instrucción siguiente (línea 90) imprime el resultado, 120.

DETECCION DE ERRORES

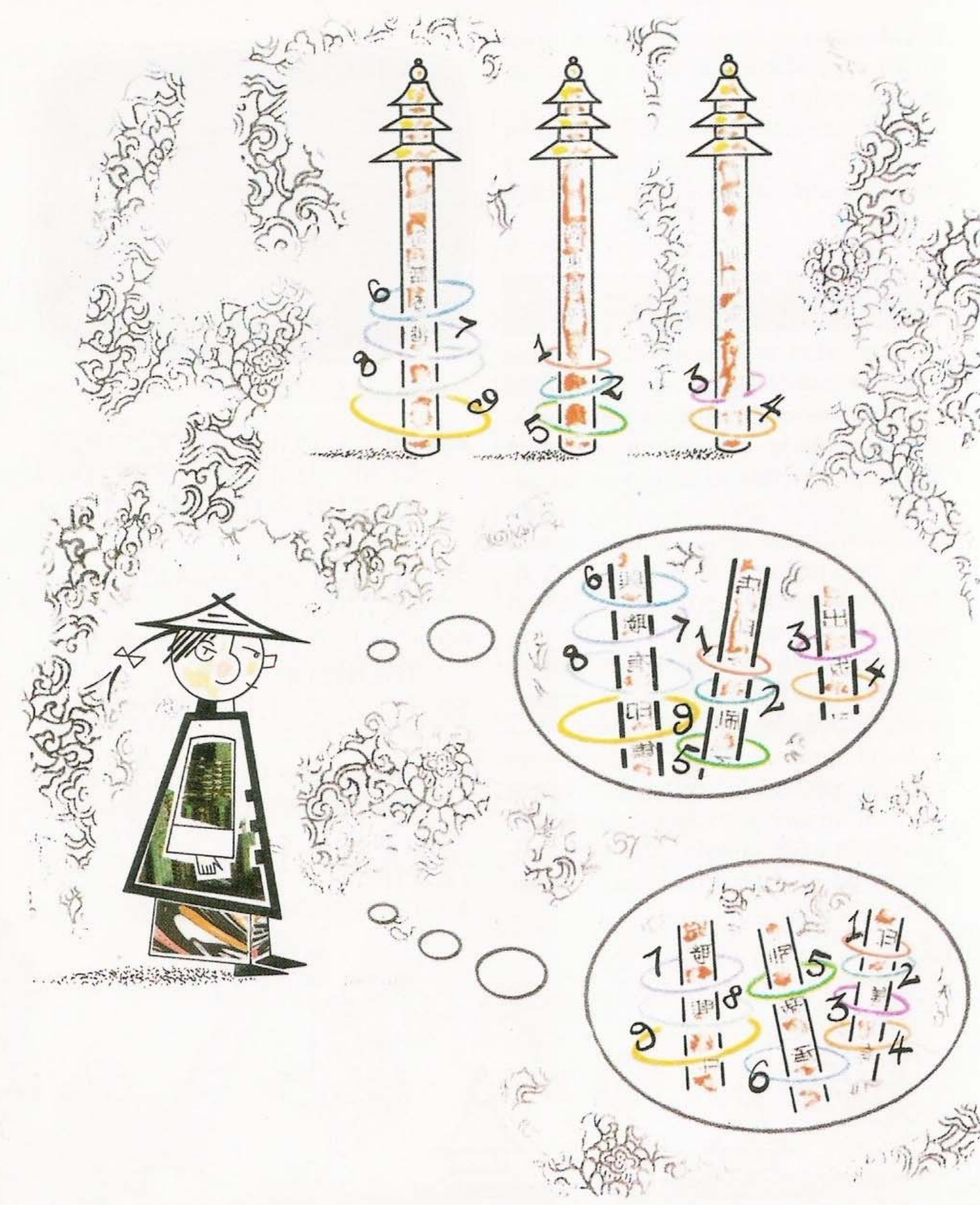
Para mantener tu programa dentro de los límites del micro y evitar catástrofes, debes saber cómo se comporta en el nivel más bajo de recursión. Habitualmente, el único medio de estar seguro de que funciona correctamente consiste en utilizar datos de comprobación de los que estés seguro. Un método simple consiste en considerar una subrutina recursiva como si fuera un grupo de copias similares de la misma subrutina. Ten en cuenta que efectuar saltos condicionales fuera de las subrutinas se considera habitualmente una mala práctica de programación, ya que la pila queda sin definir.

Cuando escribas este tipo de subrutinas, empieza siempre por una comprobación de salida efectiva. Dicha



comprobación se utiliza para decidir si el problema (tal como viene determinado por los parámetros de entrada) puede resolverse directamente, sin necesidad de subdivisión.

Antes de que empieces a programar, planea lo que quieres conseguir con la subrutina. A continuación, cuando estés escribiendo el programa, no te preocupes demasiado por la secuencia exacta de ejecución, y considera en cambio los dos principios principales, la condición de parada y la subdivisión en problemas más sencillos. Siguiendo este método, deberías ser capaz de programar alguna de las utilidades prácticas de la recursión. Veamos a continuación un ejemplo de cómo la recursión puede mejorar en gran medida la claridad y la eficiencia de un programa de ordenación.



```
1Ø PRINT '[SHIFT+CLR/HOME][
   CRSR ABAJO]>[CRSR DCH
   A.][CTRL+9]ORDENACION
    RAPIDA'
3Ø PRINT '[CRSR
   ABAJO]CUANTOS NUM.
   DESEAS ORDENAR':INPUT
   '(1-300)[CRSR DCHA.]';A
4Ø IF A< OR A>3ØØ THEN
   RUN
5Ø DIM A(A),R(1+SQR(A))
6Ø A(A)=1ØØ:PRINT '[CRSR
   DCHA.][CRSR ABAJO][CTRL+
   9]TABLA DESORDENADA:-'
   :FOR K-Ø TO A-1:A(K)=IN
   T (RND(1)*99)
65 PRINT A(K);:NEXT K:PRINT
7Ø L=ØR=A-1:GOSUB 1ØØØ
8Ø PRINT '[CRSR DCHA.][CRSR
   ABAJO][CTRL+9]TABLA OR
   DENADA:-':FOR K=Ø TO A
   -1:PRINT A(K);:NEXT K
9Ø GET Z$:IF Z$<>' ' THEN9Ø
1ØØ RUN
1ØØØ IF R>L THENI=L:J=R+1:
     V=A(L):GOTO 1Ø1Ø
1ØØ5 RETURN
1Ø1Ø I=I+:IF A(1)<V
     THEN1Ø1Ø
1Ø2Ø J=J-1:IF A(J)>V
     THEN1Ø2Ø
1Ø3Ø IF J>=I THENT=A(I)=A(J
     ):A(J) =T:GOTO 1Ø1Ø
1Ø4Ø T=A(L):A(L)=A(J):A(J)=T
1Ø5Ø R(LV)=R:LV=LV+1:R=J-1
     :GOSUB 1ØØØ
1Ø6Ø LV=LV-1:R=R(LV):L=I:GO
     SUB 1ØØØ
1Ø7Ø RETURN
```

El programa te permite introducir la cantidad de números aleatorios que deseas ordenar. La línea 50 dimensiona una matriz A (A), para almacenar dichos números, y otra matriz, R, para almacenar las variables para las llamadas recursivas. La línea 60 genera e imprime los números aleatorios desordenados, mientras que la línea 70 llama la subrutina recursiva que los pone por orden ascendente. La línea 80 imprime la tabla de números.

El método utiliza algunos aspectos de la intercalación de listas a partir de dos subconjuntos y algunos aspectos de la ordenación de sublistas. La lista principal se divide en dos subconjuntos (líneas 1010 y 1020). Observa la comprobación de salida crucial (línea 1000) para determinar cuándo debe terminar la recursión. Cada uno de los dos subconjuntos se ordena en una pasada (línea 1030). Vuelven a dividirse los subconjuntos, pero dos de ellos se juntan formando uno de los nuevos subconjuntos y, a continuación, cada uno de dichos nuevos subconjuntos se ordena en la pasada siguiente. Luego

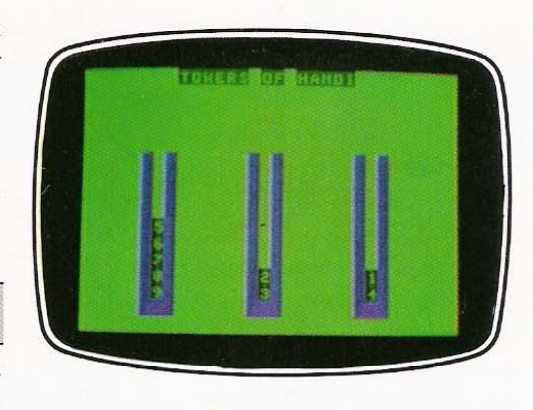
la subrutina se llama a sí misma (línea 1050) repetidamente hasta completar la ordenación.

Aunque este método de ordenación en BASIC no es tan rápido como en código máquina, es extremadamente rápido.

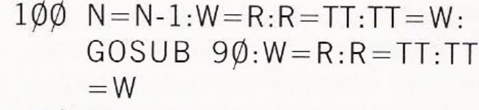
RANGO AMPLIADO

La utilidad de la recursión llega más allá del cálculo de factoriales y de otras aplicaciones matemáticas. La recursión puede ser extremadamente útil en programas de juegos y para generar figuras gráficas complicadas. También puede aplicarse esta técnica a la inteligencia artificial (AI – Artificial Intelligence), por ejemplo tanto en control de robots como en programas de juegos. Métodos similares se usan también en el proceso de lenguajes (compiladores e intérpretes).

En el ajedrez y en los programas de estrategia se utiliza la recursión. Introduce el siguiente programa para ver una ilustración simple de dicha aplicación en el problema clásico de las Torres de Hanoi.




```
2Ø PRINT '[SHIFT+CLR/HOME][CRSR ABAJO]>[CRSR DCHA.][CTRL+9]TORRES DE HANOI'
3Ø PRINT '[CRSR ABAJO]NUM. DE ANILLOS (2-9)'
4Ø INPUT N:IF N<2 OR N>9 THENRUN
5Ø TT=2:TF=1:R=3:GOSUB 9Ø
6Ø PRINT '[CRSR ABAJO]MOVIMIENTOS REALIZADOS=';2↑N-1
7Ø END
9Ø IF N=Ø THENRETURN
```



```
1ØØ N=N-1:W=R:R=TT:TT=W:GOSUB 9Ø:W=R:R=TT:TT=W
11Ø PRINT 'ANILLO';N;'DE TORRE';TF;'HASTA TORRE';TT
12Ø W=R:R=TF:TF=W:GOSUB 9Ø:W=R:R=TF:TF=W
13Ø N=N+1:RETURN
```

El problema tradicional consta de tres palos montados sobre una superficie. En el primer palo se ensartan un cierto número de discos de diámetro variable. El objeto consiste en transferir la pila de discos de uno a otro palo. Los discos pueden moverse sólo de uno en uno y ningún disco puede quedar nunca sobre otro menor que él. El tercer palo se utiliza como lugar de paso mientras se mueven los discos. La versión para ordenador da una representación gráfica del problema. Ejecuta el programa e introduce el número de discos que deseas manejar. La transferencia de discos es rápida, por lo que no serás capaz de seguir cada movimiento a menos que modifiques el programa para retardarlo del modo siguiente.

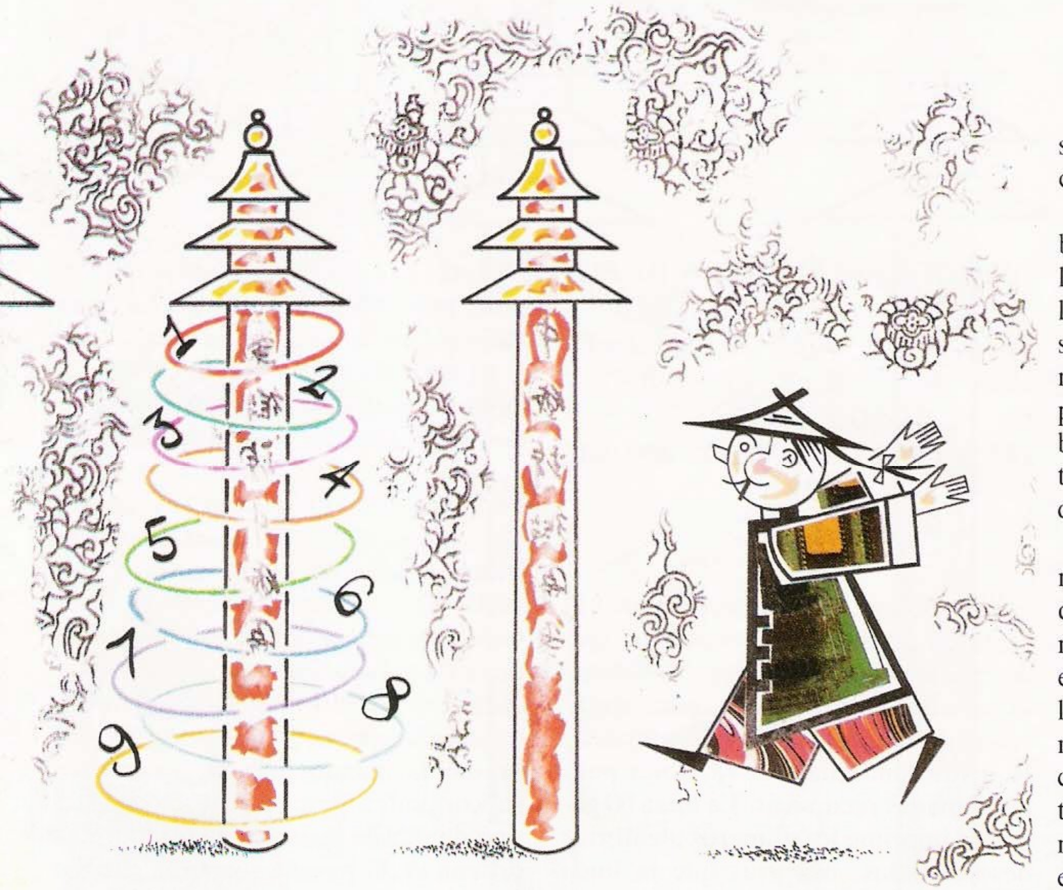
Inserta una nueva línea:

```
115 POKE 198,0:WAIT 198,1
```

Este cambio hace que el programa se ejecute sólo cuando pulses una tecla.

El programa que resuelve el problema está mejor planteado para utilizar la recursión. El formato es similar al de los programas anteriores. La subrutina recursiva imprime sucesivamente cada movimiento de uno a otro palo hasta que se completa el problema. El número total de movimientos efectuados se calcula y se visualiza después de cada jugada.

Por tanto, para problemas altamente complicados que requieran subdivisión, la recursión a menudo es el mejor método de programación. Sin embargo, si el espacio de memoria es limitado, la recursión puede ser extremadamente costosa, tanto en tiempo como en espacio. Pero si una subrutina o procedimiento se llama a sí mismo más de dos veces, la recursión es quizás el mejor método a utilizar.



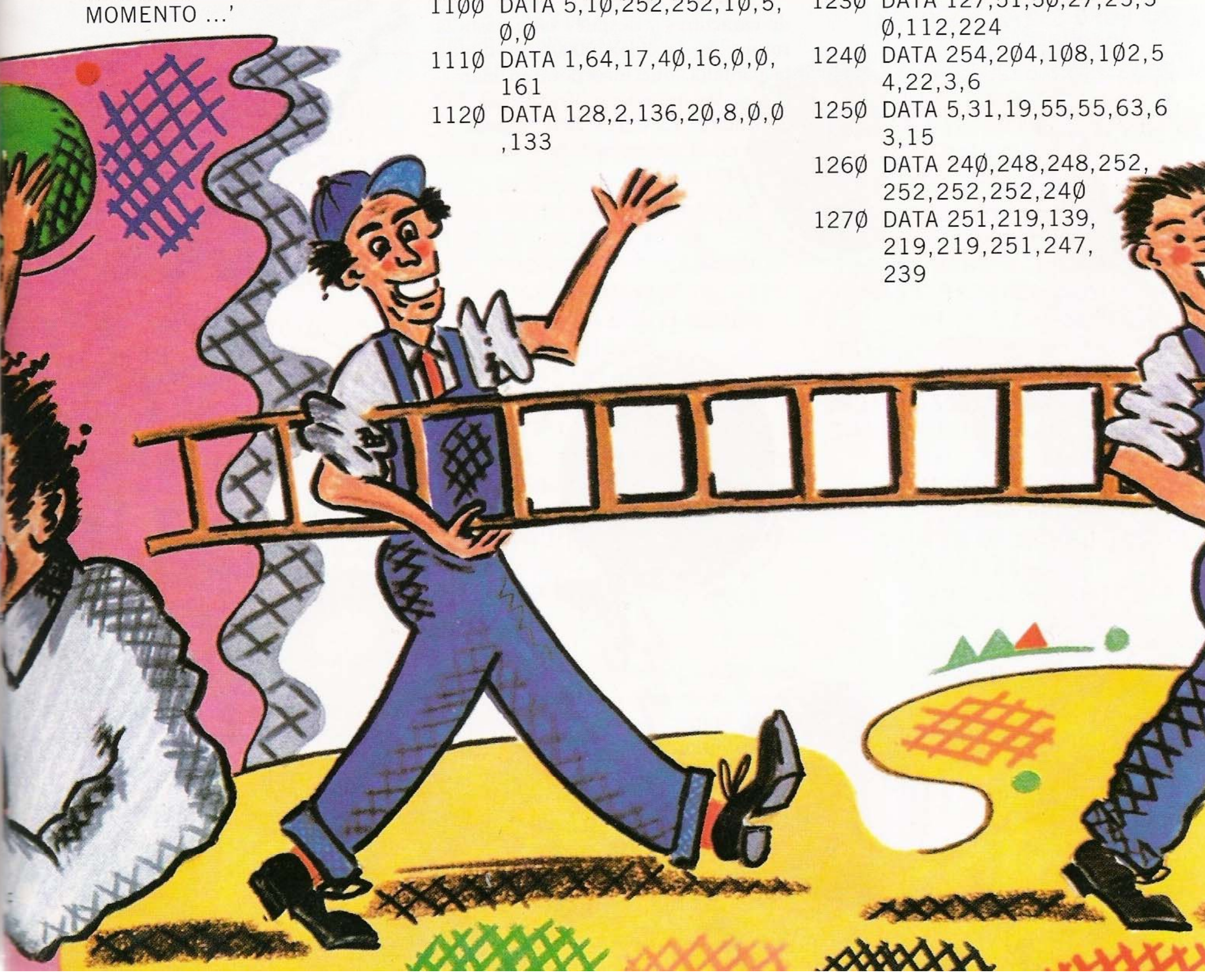
animación resulte más realista). Por último están la forma del globo y el dibujo de un globo que explota. Con dos caracteres describirás la flecha y con otros dos, la escalera. Esto da un total de 26 caracteres.

En esta primera parte tienes establecidos los gráficos e inicializado el juego.

El programa para el Commodore difiere de los de otros ordenadores como el Spectrum. Trataremos de algunos aspectos.

INICIALIZACION DE LOS GRAFICOS

```
10 POKE 51,255:POKE 52,47::P
   OKE 55,255:POKE 56,47:
   CLR:PRINT'[SHIFT+CLR/HO
   ME]ESPERAR UN
   MOMENTO ...'
15 GOSUB 1000
999 GOTO 999
1000 POKE 56 334,0:POKE 1,
     51:FOR I=0 TO 64*8-1 :P
     OKE 1228 8+I,PEEK(53
     248+I):NEXT I
1010 POKE 1,55:POKE 56334,1
1020 FOR Z=13312 TO
     13527:READ X:POKE Z,
     X:NEXT Z:RETURN
1050 DATA 15,63,127,255,255,
     255,255,127
1060 DATA 240,252,254,255,
     255,255,255,254
1070 DATA 127,63,63,31,15,7,
     3,6
1080 DATA 254,252,252,248,2
     40,224,192,96
1090 DATA 32,96,255,255,96,
     32,0,0
1100 DATA 5,10,252,252,10,5,
     0,0
1110 DATA 1,64,17,40,16,0,0,
     161
1120 DATA 128,2,136,20,8,0,0
     ,133
1130 DATA 161,0,0,16,40,17,6
     4,1
1140 DATA 133,0,0,8,20,136,2
     ,128
1150 DATA 48,48,48,48,111,1
     11,48,48
1160 DATA 12,12,12,12,246,2
     46,12,12
1170 DATA 7,31,49,57,127,11
     2,237,255
1180 DATA 224,248,140,204,2
     54,14,183,255
1190 DATA 127,59,51,99,115,
     35,6,12
1200 DATA 254,108,102,51,49
     ,25,24,48
1210 DATA 7,31,49,51,127,11
     2,235,255
1220 DATA 224,248,140,156,2
     54,14,215,255
1230 DATA 127,51,50,27,25,5
     0,112,224
1240 DATA 254,204,108,102,5
     4,22,3,6
1250 DATA 5,31,19,55,55,63,6
     3,15
1260 DATA 240,248,248,252,
     252,252,252,240
1270 DATA 251,219,139,
     219,219,251,247,
     239
```





```
1280 DATA 252,254,254,254,2
     54,254,254,252
1290 DATA 95,127,31,31,31,6
     3,127,127
1300 DATA 188,188,188,188,1
     88,124,248
1310 DATA 239,239,239,0,247
     ,247,247,0
```

Esta rutina contiene todos los DATA necesarios para establecer los gráficos. Los UDG se establecen para Freddy, la araña, la escalera y las plataformas de la parte superior e inferior de la pantalla gracias a un POKE en la memoria, después de que la línea 10 ha preparado (*clear*) el espacio suficiente.

Y AHORA LA PANTALLA

```
20 POKE 53272,28:GOSUB
   2000
2000 POKE 53280,0:POKE 532
     81,0:PRINT [SHIFT+CL
     R/HOME][CTRL+4][CRSR
      DCHA.]L=[CRSR DCHA.
     ]0[ 3*CRSR DCHA.][B=[
     CRSR DCHA.]20[ 5*CRS
     R DCHA.]SC=[CRSR DC
     HA.]0[ 6*CRSR DCHA.]
     HS='HS'[CRSR ABAJO]'
2010 FOR Z=1 TO 2:PRINT '[CT
     RL+9][COMM.+1]ZZZZZZ
     ZZZZZZZZZZZZZZZZZZZZZ
     ZZZZZZZZZZZZZZ[ 2*CRS
     R ABAJO][CTRL+0]';:NE
     XT Z
2015 PRINT '[CLR/HOME][
     3*CRSR ABAJO]';
2020 FOR Z=1 TO 2:PRINT
     '[CTRL+2][ 11*CRSR
     DCHA.])[ 11*CRSR
     DCHA.])[ 11*CRSR
      DCHA.])[ 2*CRSR
       DCHA.])':NEXT Z
2030 PRINT TAB(36)'[CTRL+8][
     CTRL+9]JK':PRINT '[CR
     SR ARRIBA]';:FOR Z=1
      TO 16:PRINT SPC(36)'
     [CTRL+8]
     [CTRL+9]JK':
     NEXT Z
2040 FOR Z=1 TO
     2:PRINT '[C
     TRL+9]
     [CRSRARR
     IBA]ZZZZZZ
     ZZZZ[COMM+5]
     ZZZZZZZZZZ[COM
     M+8]ZZZZZZZZZZ
     [CTRL+2]ZZZZZZZ
     ZZZ[CTRL+0]';
2050 NEXT Z:RETURN
```

La línea 20 conecta al nuevo juego de caracteres y después salta hasta la rutina de la línea 2000. Se dibuja la pantalla, que incorpora un marcador y las posiciones superiores del marcador. La línea 2010 dibuja la jaula de la araña, la 2030 la escalera, y la 2040 el suelo.

FREDDY Y LA ARAÑA DE MARTE (Y II)

- CULMINACION DEL JUEGO
- EL BUCLE PRINCIPAL
- FREDDY COMO DESAYUNO
- ANIMACION DE LA ARAÑA
- ESTALLIDO DE LOS GLOBOS

El juego está inicializado. Freddy espera en la escalera. Sus flechas son aguzadas, los globos están inflados y a la araña se le hace la boca agua.

En la primera parte de *Freddy y la araña de Marte* introdujiste las rutinas de inicialización y de gráficos. Ahora reúnelas añadiendo las rutinas de animación.

La introducción de estas rutinas completará el juego.

PREPARACION DE LAS VARIABLES

```
3Ø B$="@A[CRSR ABAJO][ 2
   *CRSR IZQDA.]BC":AR$="
   DE":E$="FG[CRSR ABAJO]
   [ 2*CRSR IZQDA.]HI":M$(
   1)="LM[CRSR ABAJO][ 2*
   CRSR IZQDA.]NO":M$(2)=
   "PQ[CRSR ABAJO][ 2*CRS
   R IZQDA.]RS"
4Ø S$="[CTRL+9][CTRL+5]TU
   [CRSR ABAJO][ 2*CRSR IZ
   QDA.]VW[CRSR ABAJO][ 2*
   CRSR IZQDA.][CTRL+3]XY"
   :L(1)=8:L(2)=2Ø:L(3)=32:
   L=Ø:M=Ø:D=1:BL=2Ø:BB
   =2Ø:SP=.3
5Ø Y$="[CLR/HOME][ 25*CRS
   R ABAJO]":F$(1)=")[CRS
   R ABAJO][CRSR IZQDA.])":
   F$(2)="([CRSR ABAJO][CR
   SR IZQDA.](":LL=Ø
6Ø SY=7:SD$="[CTRL+9][CTR
   L+8]JK[CRSR ABAJO][ 2*C
   RSR IZQDA.]JK[CRSR ABAJ
   O][ 2*CRSR IZQDA.]JK":K
   K=Ø:SC=Ø:F=Ø
7Ø GOTO 9Ø
```

Estas líneas simplemente inicializan las variables y cadenas necesarias.

EL GLOBO SUBE

```
8Ø FOR Z=15 TO Ø STEP-
   1:POKE 646,Z
85 PRINT LEFT$(Y$,BY)SPC (BX)
   "[CTRL+9]"E$:NEXT Z:IF
   KK=1 THENKK=Ø:
   GOTO 98
9Ø L=L+1:PRINT
   "[CLR/HOME][CTRL+4][
   3*CRSR DCHA.]"; L
95 IF L=4 THENL=3:LL=1:L(
   3)=L(3)+3
98 BX=INT (RND(1)
   *25)+1:BY=21
99 POKE 198,Ø:IF L=1
   THEN15Ø
125 PRINT LEFT$(Y$,
    BY)SPC(BX)
    "[CTRL+9][CTRL+1]"B$
13Ø BY=BY-SP:IF BY<8
    THENPRINT "[CLR/HOME][
    3*CRSR ABAJO][CTRL+1]"
    SPC(L(L)+3F$(1):GOSUB
    6ØØØ:GOTO 8Ø
135 PRINT LEFT$(Y$,
    BY)SPC(BX)
    "[CTRL+9][CTRL+4]"B$
6ØØØ BL=BL-1:IF BL<Ø THE
     NBB=BB+5:BL=BB:SP=
     SP+.3
6Ø1Ø PRINT"[CLR/HOME]
     [CTRL+4]"SPC
     (1Ø)BL"[CRSR IZQDA.]
     ":RETURN
```

Hay cuatro rutinas separadas que tienen relación con los globos. Las líneas 80 y 85 forman la rutina de estallido del globo, y se utilizan cuando la flecha da en el blanco. Las líneas 90 a 99 determinan el nivel y la posición del globo. Las líneas 125 a 135 animan los globos mientras flotan por la pantalla. Las líneas 6000 a 6010 ponen en movimiento el globo siguiente (una vez que el anterior ha estallado o ha llegado a la jaula) e incrementan la velocidad del mismo.

FREDDY SE MUEVE

```
138 IF F=Ø THENPRINT
    LEFT$(Y$,SY+1)
    SPC(35)" "
14Ø GET A$:IF A$="" THEN18Ø
15Ø IF A$="[CRSR ABAJO]"
    AND SY>8 THENGOSUB
    4ØØØ:SY=SY-1
```





```
16Ø IF A$="[CRSR DCHA.]"
    AND SY<2Ø THENGOSUB
    4ØØØ:SY=SY+1
165 PRINT LEFT$(Y$,
    SY)SPC(36)S$
17Ø IF A$=" " AND F=Ø
    THENF=1:XX=34:YY=SY+1
18Ø IF F=Ø THENPRINT LEFT$(
    Y$,SY+1) SPC(35)"[CTRL
    +9][CTRL+2]D"
9ØØ GOTO 1ØØ
4ØØØ PRINT LEFT$(Y$,SY)
     SPC(36)SD$:RETURN
```

Las líneas 138 a 180 mueven a Freddy como respuesta a las teclas pulsadas por el jugador. Las líneas 150 y 160 leen las dos teclas del cursor, utilizadas para controlar el movimiento hacia arriba y hacia abajo, mientras que la línea 170 comprueba si se ha pulsado la barra espaciadora. Si ha sido así, se lanza una flecha y se activa el indicador de disparo, F.

Si el jugador utiliza las teclas de control del cursor para mover a Freddy, se llama la rutina de borrado del hombre de la línea 4000, obteniéndose la animación del mismo.

PASE AL SALON

```
1ØØ PRINT "[CLR/HOME][
    3*CRSR ABAJO][CTRL+1][
    CTRL+9]"SPC(M)M$(RND(
    1)*2+1)
1Ø3 IF M=36 THEN3ØØØ
1Ø5 M=M+D:IF M<1 OR
    M>L(L) THEND=-D:GOTO
    1ØØ
11Ø PRINT "[CLR/HOME][
    3*CRSR ABAJO][CTRL+6][
    CTRL+9]"SPC(M)M$(RND(
    1)*2+1)
12Ø PRINT "[CLR/HOME]
    [CTRL+9][ 3*CRSR
    ABAJO][CTRL+2]"SPC
    (L(L)+3) F$ (RND(1)*2+1)
122 IF F=1 THEN5ØØØ
123 IF LL=1 THEN138
```



Las líneas 100 a 120 animan la araña, haciendo que deambule por su jaula. La línea 122 llama la rutina de animación de la flecha si el indicador de disparo ha sido activado, mientras que la línea 123 hace que el programa salte a la rutina de movimiento del globo. Manipulando el valor de LL, puede regularse la velocidad del globo, o mejor la frecuencia con la que se llama la rutina de movimiento del globo.

TODO UN CARCAJ

```
5ØØØ PRINT LEFT$(Y$,YY)SPC(X
     X)"[CTRL+1][CTRL+9]"A
     R$
5ØØ5 IF YY<2Ø THENYY=YY+.1
5Ø1Ø XX=XX-1:IF XX<Ø
     THENF=Ø:GOTO 123
5Ø2Ø PRINT LEFT$(Y$,YY)SPC
     (XX) "[CTRL+2][CTRL+9]"
     AR$
5Ø3Ø IF (XX=BX OR XX+1=BX)
     AND (INT (BY)=INT (YY)
     OR INT (BY+1)=INT (YY))
     THEN5Ø5Ø
5Ø4Ø GOTO 123
5Ø5Ø PRINT LEFT$(Y$,YY)SPC(X
     X)"[CTRL+1][CTRL+9]"A
     R$
5Ø55 F=Ø:KK=1:SC=SC+INT ((
     26-YY)/2):PRINT "[CLR/
     HOME][CTRL+4]"SPC(21
     )SC
5Ø6Ø GOSUB 6ØØØ:GOTO 8Ø
```

Esta rutina anima la flecha borrándola de la última posición y volviéndola a imprimir en la nueva posición. Además, el programa Commodore tiene en cuenta la gravedad.

¿COMIDA RAPIDA?

```
3ØØØ IF F=Ø THENPRINT LEFT$
     (Y$,SY+1)SPC(35)" "
3ØØ5 FOR Z=7 TO 21
3Ø1Ø PRINT LEFT$(Y$,Z-1)SPC
     (M) "[CTRL+8][CTRL+9]
     JK"
3Ø2Ø PRINT LEFT$(Y$,Z)CHR$(
     153)"[CTRL+9]"SPC(M)
     M$(RND(1)*2+1):FOR
     ZZ=1 TO 2Ø:NEXT ZZ,Z
3Ø3Ø IF SC>HS THENHS=SC
3Ø4Ø PRINT "[CRSR ABAJO][
     2*CRSR DCHA.][CTRL+6]
     ESTAS MUERTO !!![ 3*
     CRSR DCHA.][CTRL+8]O
     TRO JUEGO (S/N)?"
3Ø5Ø GET A$:IF A$="S"
     THEN2Ø
3Ø6Ø IF A$<>"N" THEN3Ø5Ø
3Ø7Ø PRINT "[SHIFT+CLR/HOM
     E]":POKE 53272,21:EN
     D
```

Las líneas 3000 a 3020 animan la araña, moviéndola escalera abajo y comiéndose a Freddy. El resto de la rutina actualiza la puntuación máxima, si es necesario, y ofrece otra partida.

ABISMO: PELIGROS Y PREMIOS (III)

Willie necesita un acantilado en el que poder realizar sus osadías. Y tú precisas saber cómo combinar las dos partes de programa que llevas escritas hasta aquí.

Ya ha llegado el momento de dibujar el acantilado que tiene que escalar Willie. Para trasladarlo a la pantalla, necesitas una gran cantidad de datos. Una vez más haremos uso de un programa en BASIC para pokear los datos en una tabla.

El siguiente programa proporciona los datos necesarios para los sprites: nubes, gaviotas, piedras, golosinas para la merienda, grietas, serpientes y el propio Willie. Tal vez te parezca que hay demasiados datos aquí, pero para tener una buena sensación de animación, tienes que dibujar los objetos móviles en varias posiciones. Los sprites se alternan entonces para dar la impresión de una acción continua.

```
10  REM *** DATAS PARA LOS
    SPRITES. ***
20  V=53248
30  S=230
40  FORI=S*64TO(S+8) *64
50  READA:POKEI,A:NEXT
60  POKEV+21,1:POKEV,100:PO
    KEV+1,100:POKEV+39,0:P
    OKE2040,237
100 PRINT"(CRL) LOS SPRITES
    YA ESTAN CREADOS.!!!"
110 END
14720 DATA 0,0,0,0,0,0,0,0,0,0
14730 DATA 0,0,0,7,128,0,7,
      6,4,0,7
14740 DATA 128,0,7,128,0,15,
      192,0,11,64
14750 DATA 0,11,64,0,11,64,0
      ,12,192,0
14760 DATA 7,128,0,3,0,0,3,0,
      0,3
14770 DATA 0,0,3,0,0,3,0,0,3,
      128
14780 DATA 0,3,192,0,0,0,0,0
      ,0,0
14790 DATA 0,0,0,0,0,0,0,7,
      128,0
14800 DATA 7,64,0,7,128,0,7,
      128,0,15
14810 DATA 192,0,11,64,0,6,
      96,0,28,112
14820 DATA 0,56,216,0,7,128,
      0,3,128,0
14830 DATA 3,192,0,31,96,0,
      31,96,0,24
14840 DATA 56,0,16,60,0,0,0,
      0,0,0
14850 DATA 0,0,0,0,0,0,0,0,0,
      0
14860 DATA 0,0,0,0,0,0,0,0,0,
      0
14870 DATA 0,0,0,0,0,0,0,0,0,
      0
14880 DATA 0,0,30,0,0,63,0,0,
      94,128
14890 DATA 0,237,192,0,243,
      192,0,243,192,0
14900 DATA 237,192,0,94,128,
      0,63,0,0,30
14910 DATA 0,0,0,0,0,0,0,0,0,
      15
14920 DATA 192,0,245,96,15,
      170,160,245,85,96
14930 DATA 255,170,224,255,
      245,224,255,255,224,
      143
14940 DATA 255,224,128,255,
      224,240,15,160,255,0
14950 DATA 32,255,240,96,
      255,255,224,15,255,
      224
```

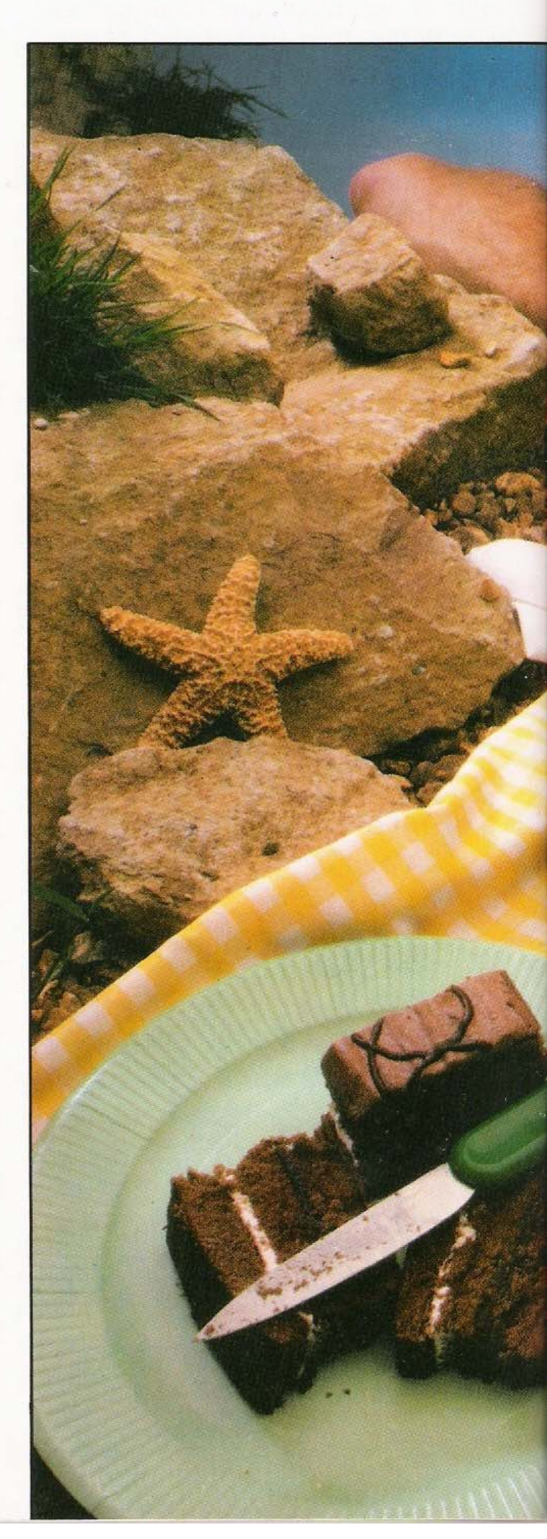


```
14960 DATA 0,255,192,
      0,15,128,0,0,0,
      0
14970 DATA 0,0,0,0,0,
      0,0,0,0,0
14980 DATA 0,0,0,0,0,
      0,0,0,0,0
```

```
14990 DATA 0,0,24,0,0,60,0,0
      ,102,0
15000 DATA 0,60,0,0,120,0,0,
      120,0,0
15010 DATA 120,0,0,60,0,0,30,
      0,0,30
15020 DATA 0,0,30,0,0,60,0,0
      ,120,0
15030 DATA 0,120,0,0,48,0,0,
      32,0,0
15040 DATA 0,36,0,0,24,0,0,8
      ,0,0
15050 DATA 16,0,0,8,0,0,24,0,
      0,60
15060 DATA 0,0,102,0,0,60,0,
      0,120,0
15070 DATA 0,120,0,0,120,0,0
      ,60,0,0
15080 DATA 30,0,0,30,0,0,30,
      0,0,60
15090 DATA 0,0,120,0,0,120,0
      ,0,48,0
15100 DATA 0,32,0,0,0,0,0,0,
      0,0
15110 DATA 0,0,0,0,0,0,0,0,0,
      0
15120 DATA 0,0,0,255,0,0,255
      ,0,0,255
15130 DATA 0,0,255,0,0,255,0
      ,0,255,0
15140 DATA 0,255,0,0,255,0,
      0,255,0,0
15150 DATA 255,0,0,255,0,0,
      255,0,0,255
15160 DATA 0,0,255,0,0,255,0
      ,0,0,0
15170 DATA 0,0,0,0,0,0,0,0,
      192,0
15180 DATA 1,32,192,2,27,48,
      26,4,8,36
15190 DATA 8,8,66,8,16,130,4
      ,96,132,0
15200 DATA 144,128,0,8,64,0,
      8,68,33,48
15210 DATA 56,65,64,8,64,
      128,7,35,0,0
15220 DATA 220,0,0,0,0,0,0,0,
      0,0
15230 DATA 0,0,0
```

El siguiente programa te proporciona los gráficos definidos por el usuario que intervienen. También se definen las letras del alfabeto. Con esto consigues una tipografía única para todo el juego. Aunque esto no es estrictamente necesario, suele hacerse con frecuencia en los juegos comerciales.

```
10 REM ** DATAS PARA LETRAS
   Y CARACTERES ESPECIALES
   **
20 FOR I=0 TO 263:READ
   A:POKE 12288+I,A:NEXT
30 PRINT "[SHIFT+CLR/HOME][
   5 ESPACIOS]ESTA
   PREPARADO UN NUEVO
   JUEGO DE"
40 PRINT "LETRAS Y ALGUNOS
   CARACTERES ESPECIALES."
50 END
12288 DATA 60,66,153,145,
      145,153,66,60
12296 DATA 56,68,130,130,
      254,130,130,0
12304 DATA 248,132,130,252,
      130,130,252,0
12312 DATA 124,130,128,128,
      128,130,124,0
12320 DATA 248,132,130,130,
      130,132,248,0
12328 DATA 248,132,128,248,
      128,132,248,0
12336 DATA 248,132,128,248,
      128,128,128,0
12344 DATA 120,132,128,152,
      132,132,120,0
12352 DATA 132,132,132,252,
      132,132,132,0
12360 DATA 56,16,16,16,16,
      16,56,0
12368 DATA 56,16,16,16,16,
      144,112,0
12376 DATA 132,136,144,224,
      144,136,132,0
12384 DATA 128,128,128,128,
      128,128,252,0
12392 DATA 130,198,170,146,
      130,130,130,0
12400 DATA 130,194,162,146,
      138,134,130,0
12408 DATA 124,130,130,130,
      130,130,124,0
12416 DATA 124,130,130,252,
      128,128,128,0
12424 DATA 124,130,130,146,
      138,134,124,0
12432 DATA 124,130,130,252,
      136,132,130,0
12440 DATA 120,132,128,120,
      4,132,120,0
12448 DATA 254,146,16,16,16,
      16,16,0
12456 DATA 130,130,130,130,
      130,130,124,0
12464 DATA 130,130,130,130,
      68,40,16,0
12472 DATA 130,130,146,146,
      146,146,124,0
```



```
12480 DATA 130,68,40,16,40,
      68,130,0
12488 DATA 130,68,40,16,16,
      16,16,0
12496 DATA 254,4,8,16,32,64,
      254,0
12504 DATA 56,56,16,254,40,
      68,130,0
12512 DATA 170,170,170,255,
      255,255,255,255
12520 DATA 3,7,15,31,31,63,
      63,127
12528 DATA 223,239,255,255,
      239,247,255,255
12536 DATA 223,239,255,255,
      239, 247,255,255
12544 DATA 0,0,0,0,0,0,0,0
```

Ejecuta estos programas y a continuación almacena la tabla de datos en código máquina que resulta, para lo cual puedes utilizar tu monitor de CM.

AGRUPANDO LAS PIEZAS

Los programas largos con frecuencia han de ser escritos y ensamblados por partes. Esto te plantea el problema de la mezcla de las distintas partes para obtener un único programa largo que funcione. Y el hecho de que todas las partes funcionen separadamente, no significa necesariamente que todo vaya a seguir funcionando al ponerlo todo junto.

Cuando una rutina llama a otra, siempre existe el problema de que la llame en el lugar inadecuado. Y también puede ocurrir que las tablas de datos no coincidan adecuadamente con los correspondientes punteros de datos. Pero estos problemas se pueden eliminar corrigiendo detenidamente los programas individuales componentes.

Esto significa que cuando hayas terminado de mezclar todas las partes del programa, no habrás perdido nada en el proceso. Por desgracia, es muy fácil perder uno o dos bytes al encadenar varios programas diferentes en la memoria. E incluso la pérdida de un solo byte puede originar que todo el programa se interrumpa o funcione mal.

Uno de los problemas más frecuentes es la sobreescritura. Si tienes algunos bytes de más, existe el peligro de que escribas algunos de ellos por encima de la rutina siguiente. Si estás almacenando las rutinas ensambladas con ayuda de tu monitor en código máquina, tienes que indicar la dirección de comienzo y el número de bytes que quieres guardar.

Con las direcciones iniciales no hay evidentemente ningún problema. Con el CM ensamblado a partir del programa en lenguaje ensamblador, la dirección de comienzo del código objeto es la misma que la de origen. Al fin y al cabo, cuando se hace actuar el ensamblador, éste va tomando las direcciones y ensamblando el programa en código máquina a partir de la dirección inicial. Y además, con las tablas de datos pokeadas a partir del BASIC, la dirección de comienzo es el valor inicial de la variable que controla el bucle FOR ... NEXT, el cual va extrayendo los datos y almacenándolos en memoria byte a byte al ejecutar el programa.

Pero el cálculo del número de bytes que hay que almacenar en memoria puede ocasionarte algún problema. La



dirección final que dan los ensambladores es con frecuencia la primera dirección libre que sigue al final del programa. Lo único que se da a veces es la dirección de comienzo de la última instrucción, por lo que si la rutina no termina con un RTS no se obtendrá una verdadera dirección final.

La manera usual de obviar este problema es almacenar un par de bytes adicionales. Pero esto puede ocasionar problemas de sobreescritura. La solución está en cargar tus programas ordenadamente en la memoria por medio del comando LOAD «nombre», 1, 1. Carga en primer lugar el que tiene el valor más bajo de dirección inicial, a continuación el siguiente que la tenga más baja, etc. De esta forma resulta que los posibles bytes adicionales que haya al final de la rutina quedarán borrados al escribirse sobre ellos parte del principio de la rutina siguiente. Y también significa que cualquer RTS que haya sido añadido con fines de prueba de las rutinas individuales, quedará igualmente borrado.

ALMACENAMIENTO

Cuando ya estén cargadas todas las rutinas y tablas de datos, almacénalas en cinta o disco como un programa bajo un nombre distinto; la dirección de comienzo será la dirección donde empieza el primer programa y la dirección final será la dirección donde termina el último programa. Si tienes algún problema ahora, puedes agrupar las distintas piezas y realizar una nueva prueba. Al utilizar la opción SAVE de tu ensamblador, no te olvides de almacenar también los programas de lenguaje ensamblador y de pokeado en BASIC, en el caso de que los vayas a necesitar para volver a ensamblar en una fase posterior.

Si tienes problemas con las tablas de datos, puedes cargar ordenadamente en la memoria las rutinas de código máquina y cargar y ejecutar nuevamente los programas de pokeado en BASIC. Para hacer esto, tienes que ir cargando los programas uno a uno, ejecutarlos y hacer un NEW antes de pasar a cargar el siguiente. También en este caso tienes que almacenar en cinta toda el área de memoria, asignándole un nuevo nombre de programa.

Ya hemos visto las presentaciones y los títulos. Ya hemos escuchado la obertura. Es tiempo de que se levante el telón. En este caso de *Abismo*, más bien que levantar el telón, tendremos que hacer un desplazamiento vertical (*«scrolling»*) de pantalla.

En realidad es un proceso bastante sencillo. Ya tienes los datos que definen el perfil de la pendiente. Por encima de la pendiente está el cielo y por debajo queda la tierra. Es muy sencillo colorear ambas secciones. Pero además tienes que desplazar la página de instrucciones y hacer que desaparezca y en su lugar aparezcan el cielo y el abismo.

Por desgracia es muy difícil hacer un scrolling en el Commodore. No obstante es muy fácil hacerlo en una escena que ya está dibujada y sólo tiene que repetirse desplazada. Como todas las escenas son básicamente iguales, no se trata más que de ir tomando lo que va desapareciendo por un extremo y hacerlo aparecer por el otro extremo. Pero hacer que aparezcan cosas nuevas en la pantalla, ya es muy diferente. Por ello lo que haremos será suprimir un nivel tan pronto como haya desaparecido la página de instrucciones.

Pero primero tienes que pintar dicho nivel. El siguiente programa en BASIC contiene los datos del acantilado. Este programa traslada dichos datos a la pantalla, de modo que puedas apreciar la silueta de la escena. Por el momento no tiene mucho sentido, ya que está en forma de cadenas de caracteres ASCII. Pero en seguida podrás ver lo que hace.

El programa BASIC almacena entonces el contenido de la pantalla en otra zona de memoria en donde pueda ir a buscarlo el código máquina. Al hacer esto, realizará la conversión a gráficos de ROM, de forma que lo que aparece entonces en pantalla se parece realmente a un acantilado.

```
5 REM ** CREA EL ESCENARIO DEL JUEGO **
10 PRINT "[SHIFT+CLR/HOME][17*CRSR ABAJO]]££[CRSR ARRIBA]]££[CRSR ARRIBA]]££[[CRSR ARRIBA]]££££[CRSR ARRIBA]]£[CRSR ARRIBA]]£££[CRSR ARRIBA]]£££[CRSR ARRIBA]]£ CRSR ARRIBA]]£[CRSR ARRIBA]]££";
20 PRINT "[CRSR ARRIBA]]£[CRSR ARRIBA]]££££[CRSR ARRIBA]]£"
30 FOR X=0 TO 39:F=0:FOR Y=0 TO 24
40 SC=1024:CL=55296
50 P=INT (RND(1)*3)+1:IF P=1 THENV=30
60 IF P=2 THENV=31
70 IF P=3 THENV=63
80 C=X+Y*40:IF F=1 THENPOKE SC+C,V:POKE CL+C,0
90 IF PEEK(SC+C)=28 OR PEEK(SC+C)=29 THEN F=1
100 NEXT Y,X
110 PRINT "[CLR/HOME][CRSR ABAJO]VIDAS[CRSR ABAJO][ 5*CRSR IZQDA.]NIVEL[CRSR ABAJO][ 5*CRSR IZQDA.]PUNTOS 000000[CLR/HOME]"TAB(27)"[CRSR ABAJO]=[ 3*CRSR DCHA.]=[CRSR ABAJO]="
120 FOR I=0 TO 999:P=PEEK(SC+I):POKE 13312+I,P:NEXT
```

```
13Ø PRINT "[CLR/HOME]TERMI
    NADO":STOP
```

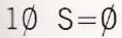
Puedes guardar estos datos con ayuda de tu programa monitor en código máquina. La tabla de datos ocupa 999 bits a partir de la posición 13312.

PINTANDO EL PAISAJE

La siguiente rutina va extrayendo los datos del paisaje y los pone en la pantalla como parte del juego:

```
63ØØ A9 ØØ       LDA #$ØØ
63Ø2 85 FB       STA $FB
63Ø4 A9 Ø4       LDA #$Ø4
63Ø6 85 FC       STA $FC
63Ø8 A9 34       LDA #$34
63ØA 85 FE       STA $FE
63ØC A9 ØØ       LDA #$ØØ
63ØE 85 FD       STA $FD
631Ø AØ ØØ       LDY #$ØØ
6312 B1 FD       LDA ($FD),Y
6314 91 FB       STA ($FB),Y
6316 2Ø 5Ø 51    JSR $515Ø
6319 E6 FB       INC $FB
631B DØ Ø2       BNE $631F
631D E6 FC       INC $FC
631F E6 FD       INC $FD
6321 DØ Ø2       BNE $6325
6323 E6 FE       INC $FE
6325 A5 FB       LDA $FB
6327 C9 E8       CMP #$E8
6329 DØ E7       BNE $6312
632B A5 FC       LDA $FC
632D C9 Ø7       CMP #$Ø7
632F DØ E1       BNE $6312
6331 6Ø          RTS
```

A continuación os presentamos su cargador en BASIC:



```
1Ø S=Ø
2Ø FOR AD=25344 TO 25393
3Ø READBY:POKEAD,
   BY:S=S+BY
4Ø NEXT
5Ø IFS<>8245 THEN
   PRINT"ERROR EN LOS
   DATOS":END
25344 DATA 169,Ø,133,251,
      169,4
2535Ø DATA 133,252,169,52,
      133,254
25356 DATA 169,Ø,133,253,
      16Ø,Ø
25362 DATA 177,253,145,251,
      32,8Ø
25368 DATA 81,23Ø,251,2Ø8,2
      ,23Ø
25374 DATA 252,23Ø,253,2Ø8,
      2,23Ø
2538Ø DATA 254,165,251,2Ø1,
      232,2Ø8
25386 DATA 231,165,252,2Ø1,
      7,2Ø8
25392 DATA 225,96
```

Ya puedes ensamblar y guardar este programa, pero no lo llames, porque desde él se produce una llamada a una subrutina que veremos más adelante y si no encuentra dicha subrutina en memoria, se interrumpirá.

PINTANDO CON LOS NUMEROS

Las primeras cuatro instrucciones establecen el puntero de pantalla y las cuatro siguientes definen otro puntero de página cero para los datos.

A continuación se pone a cero el desplazamiento de Y, ya que vas a incrementar la página cero. El byte al que señala el puntero de datos es cargado en el acumulador y almacenado en la posición de pantalla señalada. Se llama entonces a la rutina que hay en $5150. Más adelante nos ocuparemos de ella que es la que define los colores.

Las seis instrucciones siguientes incrementan los punteros. Las instrucciones BNE de esta parte de la rutina comprueban si el byte bajo del puntero ha llegado al final de la página y fuerzan el salto subsiguiente a INC, que incrementa el byte alto, en caso de que no se haya llegado al final de página. Las seis instrucciones que figuran a continuación comprueban si se ha llegado o no al final de la pantalla, saliendo de la pantalla en el caso de que así sea. La instrucción RTS quedará borrada cuando se añada el resto del programa.

PONIENDOLE COLOR

Con la siguiente rutina se introducen los colores:

```
515Ø AØ ØØ       LDY #$ØØ
5152 B1 FB       LDA ($FB),Y
5154 8D 84 Ø3    STA $Ø384
5157 A5 FC       LDA $FC
5159 18          CLC
515A 69 D4       ADC #$D4
515C 85 FC       STA $FC
515E AD 84 Ø3    LDA $Ø384
5161 C9 1C       CMP #$1C
5163 FØ 1C       BEQ $5181
5165 C9 1D       CMP #$1D
5167 FØ 18       BEQ $5181
5169 C9 3F       CMP #$3F
516B FØ 19       BEQ $5186
516D C9 1E       CMP #$1E
516F FØ 15       BEQ $5186
5171 C9 1F       CMP #$1F
5173 FØ 11       BEQ $5186
5175 A9 Ø2       LDA #$Ø2
5177 91 FB       STA ($FB),Y
5179 A5 FC       LDA $FC
517B 38          SEC
517C E9 D4       SBC #$D4
517E 85 FC       STA $FC
518Ø 6Ø          RTS
```

```
5181 A9 Ø5       LDA #$Ø5
5183 4C 77 51    JMP $5177
5185 A9 Ø1       LDA #$Ø1
5188 4C 77 51    JMP $5177
```

El cargador BASIC será:

```
1Ø S=Ø
2Ø FOR AD=2Ø816 TO 2Ø874
3Ø READBY:POKEAD,BY:S=S+B
   Y
4Ø NEXT
5Ø IFS<>7678 THEN
   PRINT"ERROR EN LOS
   DATOS":END
2Ø816 DATA 16Ø,Ø,177,251,
      141,132
2Ø822 DATA 3,165,252,24,1Ø5
      ,212
2Ø828 DATA 133,252,173,132,
      3,2Ø1
2Ø834 DATA 28,24Ø,28,2Ø1,29
      ,24Ø
2Ø84Ø DATA 24,2Ø1,63,24Ø,25
      ,2Ø1
2Ø846 DATA 3Ø,24Ø,21,2Ø1,31
      ,24Ø
2Ø852 DATA 17,169,2,145,251
      ,165
2Ø858 DATA 252,56,233,212,
      133,252
2Ø864 DATA 96,169,5,76,119,
      81
2Ø87Ø DATA 169,1,76,119,81
```

La subrutina empieza cargando el mismo byte de datos que figuraba en el programa principal y almacenándolo en la dirección $0384. Se trata de un almacenamiento temporal, ya que en este momento hace falta utilizar el acumulador para otras cosas.

Las cuatro instrucciones siguientes suman el valor $D4 al byte alto del puntero de pantalla contenido en $FC. Esto desplaza el puntero desde su posición en la pantalla gráfica —que empieza en $0400— a su correspondiente posición en la pantalla de color, que empieza en $D800.

Seguidamente los datos se cargan de nuevo desde $0384 al acumulador. Se realiza entonces una comprobación de los diferentes códigos de control de los datos. Dependiendo del código de control que se encuentre, el procesador salta a la instrucción que hay en $5181 o $5186.

La instrucción que hay en $5181 es LDA #$05 que carga el acumulador con una tinta de color verde. Por el contrario la instrucción LDA #$01, que figura en $5186, asigna para la tinta el color blanco. Así es como se obtienen el color verde de la hierba y el blanco del propio acantilado. Los dos tonos de la hierba y el acantilado se obtienen con ayuda de los caracteres gráficos que mezclan el color de la tinta con el del papel.

Si no se extrae ningún carácter de control, el acumulador se carga con el color rojo. De esta forma, todo lo que hay en la pantalla que no es ni verde ni blanco, se colorea en rojo.

Naturalmente esto no significa que el cielo sea rojo. Lo que es rojo es el color de la tinta, por lo que sólo aparece cuando se escriben datos en la pantalla. En consecuencia, el número de vidas, la altura y la puntuación se presentan en color rojo, pero el cielo aparece en el color del papel, que es el gris. Por desgracia estaba bastante nublado el día que Willie decidió irse de merienda campestre, al menos en la versión Commodore.

Cuando el carácter se imprime en blanco o en verde, el procesador salta a la instrucción que hay en $5177, que es la que establece el color rojo para el resto de los datos. Así se almacena en el lugar adecuado el color elegido en la pantalla de color.

Las cuatro instrucciones siguientes restan el valor $D4 del byte alto del puntero de $FC, a fin de desplazarlo desde la pantalla de color a la pantalla de gráficos. Cuando con la instrucción RTS se retorna a la rutina principal, puede entonces extraerse el siguiente carácter gráfico y presentarse en la pantalla.

Una vez formada la primera pantalla, las siguientes son simplemente modificaciones de la misma. En el segundo nivel, Willie tiene que enfrentarse con las grietas. Para crearlas, basta con escribir por encima de la pendiente que figuraba en la primera pantalla, utilizando el color de fondo



y con la forma que quieras darle a la oquedad que vas a crear. A continuación se añaden las serpientes escribiendo con el color de fondo. Posteriormente añadiremos la rutina que hace que se mueva la serpiente.

La siguiente rutina determina qué sprites aparecen en cada nivel e inicializa la posición de comienzo del hombre y el canto rodado. También se ocupa del mar.

```
6ØØØ 4C 59 6Ø    JMP $6Ø59
6ØØ3 AD ØØ CØ    LDA $CØØØ
6ØØ6 C9 Ø1       CMP #$Ø1
6ØØ8 DØ Ø6       BNE $6Ø1Ø
6ØØA A9 47       LDA #$47
6ØØC 8D 15 DØ    STA $DØ15
6ØØF 6Ø          RTS
6Ø1Ø C9 Ø2       CMP #$Ø2
```



```
6Ø12 DØ 18       BNE $6Ø2C
6Ø14 A9 7D       LDA #$7D
6Ø16 8D 15 DØ    STA $DØ15
6Ø19 A2 Ø3       LDX #$Ø3
6Ø1B AØ ØØ       LDY #$ØØ
6Ø1D A9 ØF       LDA #$ØF
6Ø1F 99 2A DØ    STA $DØ2A,Y
6Ø22 A9 EC       LDA #$EC
6Ø24 99 FB Ø7    STA $Ø7FB,Y
6Ø27 C8          INY
6Ø28 CA          DEX
6Ø29 DØ F2       BNE $6Ø1D
6Ø2B 6Ø          RTS
6Ø2C C9 Ø3       CMP #$Ø3
6Ø2E DØ Ø8       BNE $6Ø38
6Ø3Ø A9 7F       LDA #$7F
6Ø32 8D 15 DØ    STA $DØ15
6Ø35 4C 19 6Ø    JMP $6Ø19
6Ø38 C9 Ø4       CMP #$Ø4
6Ø3A DØ 18       BNE $6054
6Ø3C A9 7D       LDA #$7D
6Ø3E 8D 15 DØ    STA $DØ15
6Ø41 A2 Ø3       LDX #$Ø3
6Ø43 AØ ØØ       LDY #$ØØ
6Ø45 A9 Ø5       LDA #$Ø5
6Ø47 99 2A DØ    STA $DØ2A,Y
6Ø4A A9 EA       LDA #$EA
6Ø4C 99 FB Ø7    STA $Ø7FB,Y
6Ø4F C8          INY
6Ø5Ø CA          DEX
6Ø51 DØ F2       BNE $6Ø45
6Ø53 6Ø          RTS
6Ø54 A9 7F       LDA #$7F
6Ø56 4C 3E 6Ø    JMP $6Ø3E
6Ø59 A9 Ø7       LDA #$Ø7
6Ø5B 8D ØE CØ    STA $CØØE
6Ø5E A9 E8       LDA #$E8
6Ø6Ø 8D ØD CØ    STA $CØØD
6Ø63 AD Ø2 CØ    LDA $CØØ2
6Ø66 8D Ø2 CØ    STA $CØØ2
6Ø69 8D ØC CØ    STA $CØØC
6Ø6C A9 Ø2       LDA #$Ø2
6Ø6E 8D 12 CØ    STA $CØ12
6Ø71 A9 21       LDA #$21
6Ø73 8D 11 CØ    STA $CØ11
6Ø76 A9 12       LDA #$12
6Ø78 8D ØØ DØ    STA $DØØØ
6Ø7B A9 A1       LDA #$A1
6Ø7D 8D Ø1 DØ    STA $DØØ1
6Ø8Ø A9 42       LDA #$42
6Ø82 8D 1Ø DØ    STA $DØ1Ø
6Ø85 A9 4A       LDA #$4A
6Ø87 8D ØD DØ    STA $DØØD
6Ø8A A9 4Ø       LDA #$4Ø
6Ø8C 8D ØC DØ    STA $DØØC
6Ø8F AD 1E DØ    LDA $DØ1E
6Ø92 AD 1F DØ    LDA $DØ1F
6Ø95 4C Ø3 6Ø    JMP $6ØØ3
```

Su consiguiente cargador en BASIC es:

```
1Ø S=Ø
2Ø FOR AD=24576 TO
   24727
3Ø READBY:POKEAD,BY:S=S+B
   Y
4Ø NEXT
5Ø IFS<>185Ø5 THEN
   PRINT"ERROR EN LOS
   DATOS":END
24576 DATA 76,89,96,173,Ø,
      192
24582 DATA 2Ø1,1,2Ø8,6,169,
      71
24588 DATA 141,21,2Ø8,96,
      2Ø1,2
24594 DATA 2Ø8,24,169,125,
      141,21
246ØØ DATA 2Ø8,162,3,16Ø,Ø,
      169
246Ø6 DATA 15,153,42,2Ø8,
      169,236
24612 DATA 153,251,7,2ØØ,
      2Ø2,2Ø8
24618 DATA 242,96,2Ø1,3,3,
      2Ø8,Ø
24624 DATA 169,127,141,21,
      2Ø8,76
2463Ø DATA 25,96,2Ø1,4,2Ø8,
      24
24636 DATA 169,125,141,21,
      2Ø8,162
24642 DATA 3,16Ø,Ø,169,5,
      153
24648 DATA 42,2Ø8,169,234,
      153,251
24654 DATA 7,2ØØ,2Ø2,2Ø8,
      242,96
2466Ø DATA 169,127,76,62,96
      ,169
24666 DATA 7,141,14,192,169
      ,232
24672 DATA 141,13,192,173,2
      ,192
24678 DATA 141,2,192,141,12
      ,192
24684 DATA 169,2,141,18,192
      ,169
2469Ø DATA 33,141,17,192,
      169,18
24696 DATA 141,Ø,2Ø8,169,
      161,141
247Ø2 DATA 1,2Ø8,169,66,141,
      16
247Ø8 DATA 2Ø8,169,74,141,
      13,2Ø8
24714 DATA 169,64,141,12,
      2Ø8,173
2472Ø DATA 3Ø,2Ø8,173,31,
      2Ø8,76
24726 DATA 3,96
```

DANDO SALTOS

Esta rutina empieza con un salto a una etiqueta situada hacia la mitad de la propia rutina. Y el procesador sólo regresa al principio una vez que se ha completado la segunda mitad. Aunque ésta no es la manera más adecuada de escribir programas, se trata de un método útil para pasar de un segmento de programa a otro sin necesidad de tener que volver a escribir todo el código fuente.

LOS SPRITES

En el curso de la escritura del juego se construye una tabla de variables. Dicha tabla comienza en 49152 y se utiliza para almacenamiento temporal de los parámetros que cambian durante el juego. La variable almacenada en la dirección 49152 especifica la altura alcanzada por el jugador y en consecuencia determina qué pantalla y qué sprites se requieren.

Los contenidos de la dirección 49152 se cargan en el acumulador y se comparan con 1. Si el juego no está en el nivel 1, la instrucción BNE fuerza un salto a la siguiente subrutina. Si se encuentra en el nivel 1, el registro A se carga con el número 71 que después se almacena en la dirección 53269. Esta es la dirección que habilita la presentación de los sprites. Los sprites se activan poniendo un 1 en el conjunto de bits. Según esto, se activan los sprites 0, 1, 2 y 6.

A continuación se produce el retorno del procesador.

EN EL NIVEL DOS

La siguiente subrutina empieza de la misma manera, excepto que esta vez la comprobación se hace para ver si se encuentra en el nivel dos. Si así ocurre, se activan además los sprites 3, 4 y 5, mientras que el sprite 2 (que corresponde al canto rodado) queda desactivado.

A continuación se carga el registro X con el valor 3 como contador de bucle, mientras que Y se carga con 0 como valor de desplazamiento que será incrementado cada vez que el procesador recorra el bucle. De esta forma se carga el valor 15, que es el color de fondo, en las direcciones de color de los sprites 3, 4 y 5. Después se activan los correspondientes punteros de sprite. La subrutina realiza los agujeros.

La siguiente subrutina comprueba si el jugador se encuentra en el nivel tres. De ser así, se activa de nuevo el sprite del guijarro, para lo cual se suma un dos adicional y se almacena en la dirección que habilita los sprites. Pero todavía hay que perforar los agujeros por lo que el procesador regresa a la rutina RET anterior.

En el nivel cuatro, el sprite del canto rodado se desactiva de nuevo y se ejecuta el nuevo bucle que empieza en la dirección $6045 (24645). Se trata de la misma rutina anterior de perforación de agujeros, con la salvedad de que el color se ha puesto a 5, que es el color de la serpiente y de que el puntero de sprites está apuntando a los datos de la serpiente.

Finalmente en el nivel cinco, el sprite del canto rodado vuelve a activarse otra vez y se llama a la rutina de presentación de serpientes.

EL MAR

Las variables contenidas en las direcciones 49165 ($COOD) y 49166 ($CODE) determinan el estado de la mar. El retardo que determina cuán rápidamente sube la agua está contenido en la dirección 49154 ($COO2) y puede cambiarse durante el juego para hacer más desesperada la escalada de Willie y poner el juego más difícil.

Se definen entonces las coordenadas X e Y de los sprites que conforman el mar y se desactivan los indicadores de colisión de sprites.

SERPIENTES Y CONTADORES

La siguiente rutina configura las serpientes:

```
61ØØ A9 74      LDA #$74
61Ø2 8D Ø6 DØ   STA $DØØ6
61Ø5 A9 92      LDA #$92
61Ø7 8D Ø7 DØ   STA $DØØ7
61ØA A9 7A      LDA #$7A
61ØC 8D Ø8 DØ   STA $DØØ8
61ØF A9 AØ      LDA #$AØ
6111 8D Ø9 DØ   STA $DØØ9
6114 A9 82      LDA #$82
6116 8D ØA DØ   STA $DØØA
6119 A2 Ø3      LDX #$Ø3
611B AØ ØØ      LDY #$ØØ
611D A9 14      LDA #$14
611F 99 64 C3   STA $C364,Y
6122 E9 Ø5      SBC #$Ø5
6124 C8         INY
6125 CA         DEX
6126 DØ F7      BNE $611F
6128 6Ø         RTS
```

A continuación os proporcionamos su cargador en BASIC:

```
1Ø S=Ø
2Ø FOR AD=24832 TO
   24872
3Ø READBY:POKEAD,BY:S=S+
   BY
4Ø NEXT
5Ø IFS<>5457 THEN
   PRINT"ERROR EN LOS
   DATOS":END
24832 DATA 169,116,141,6,
      2Ø8,169
24838 DATA 146,141,7,2Ø8,
      169,122
24844 DATA 141,8,2Ø8,169,
      16Ø,141
2485Ø DATA 9,2Ø8,169,13Ø,
      141,1Ø
24856 DATA 2Ø8,162,3,16Ø,Ø,
      169
24862 DATA 2Ø,153,1ØØ,195,
      233,5
24868 DATA 2ØØ,2Ø2,2Ø8,247,
      96
```

Las coordenadas X e Y de los tres agujeros y serpientes se cargan en el acumulador y se almacenan en las direcciones de memoria 53254 ($DOO6) a 53258 ($DOOA). Estas son las direcciones que se encargan de controlar las posiciones X e Y de los sprites 3, 4 y 5.

El bucle escalona los movimientos de las lenguas, de forma que no se disparen todas al mismo tiempo.



¡NO TE PIERDAS NI UN SOLO EJEMPLAR!

INPUT COMMODORE quiere proporcionar a sus lectores este nuevo servicio de ejemplares atrasados para que no pierdan la oportunidad de tener en sus hogares todos los ejemplares de esta revista, líder en el mercado español.

Podréis solicitar cualquier número de **INPUT COMMODORE** que querais, siempre al precio de cubierta (sin más gastos).

Utiliza el cupón adjunto, enviándolo a **EDISA** (Dpto. de Suscripciones), López de Hoyos, 141 - 28002 Madrid.

1 2 3 4 5 6 7 8 9 10 11 12 13

CUPON DE PEDIDO

SI, envíenme contrarreembolso .... ejemplares de **INPUT COMMODORE** de los números:

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

NOMBRE

APELLIDOS

DOMICILIO

NUM. PISO ESCALERA COD. POSTAL

POBLACION PROV.

TELEFONO FIRMA

siempre a tu servicio

ANIMACION MEDIANTE GRAFICOS PAGINADOS

La utilización de técnicas simples de gráficos paginados ofrece una real comprensión del mundo de la animación por ordenador y obtiene unos resultados interesantes en tu micro.

Todos los tipos de animación se basan en un fenómeno de percepción conocido como persistencia de la visión. En efecto, esto significa que una imagen que vemos se «conserva» en la memoria durante un instante apreciable, incluso aunque la visión se mueva hacia cualquier otro punto. Si se muestra rápidamente una secuencia de imágenes estáticas, el cerebro no puede retener los cambios de imagen que ocurren a una frecuencia superior a doce veces por segundo, aproximadamente. Como resultado, deja de separar las imágenes y, confundido, cree ver movimiento.

El proceso se demuestra con mucha sencillez con el «cambio de página», en el cual los dibujos de cada una de las páginas de un libro pueden ponerse en movimiento a medida que se pasan las páginas a una cierta velocidad. Una demostración más sofisticada de lo mismo se encuentra en el proceso de animación por cuadros, que data de los primeros años de este siglo.

Este tipo de animación consiste en dibujar una figura sobre una pieza de plástico, denominada *celda*, fotografiando a continuación un par de cuadros de película utilizando una cámara de cine convencional montada sobre una montura situada sobre dicha película. La celda se sustituye a continuación por otra que muestra una versión ligeramente alterada de la misma figura y se repite todo el proceso anterior. Como puedes imaginar, este tipo de animación requiere un increíble número de imágenes que deben dibujarse concienzudamente a mano, ya que se necesitan alrededor de unos veinticinco cuadros por cada segundo de película cinematográfica acabada.

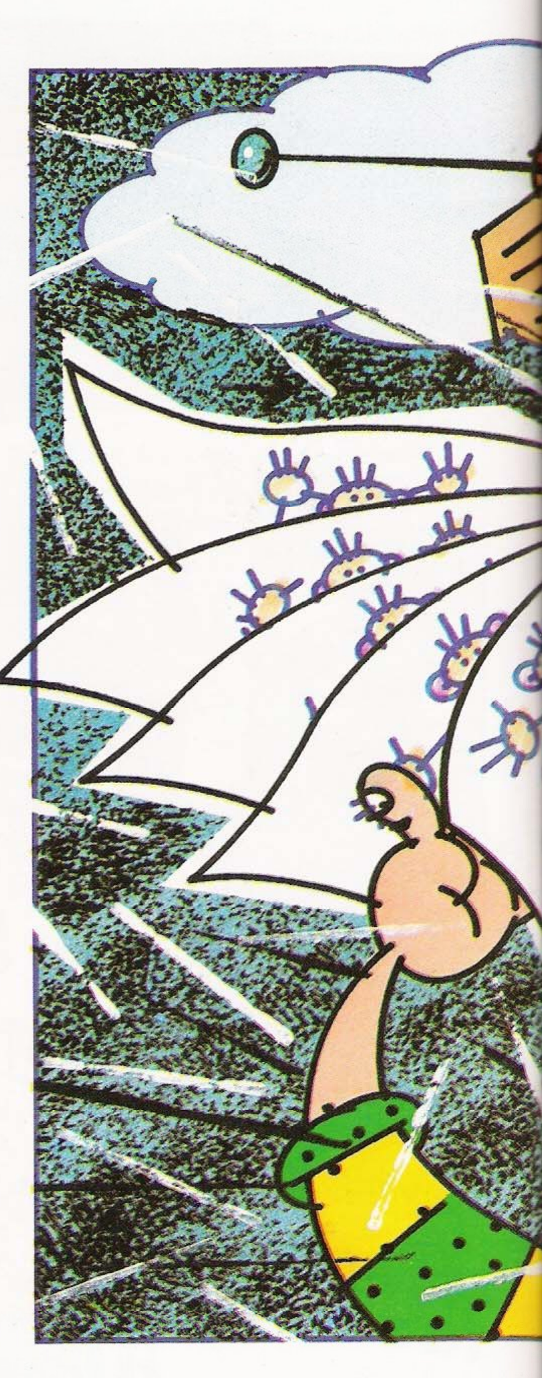
GRAFICOS POR ORDENADOR

Por tanto, ¿por qué no utilizar los ordenadores para acelerar el proceso? Incluso el relativamente humilde micro doméstico puede generar buenas imágenes, mientras que los ordenadores especiales son capaces de obtener imágenes asombrosamente elaboradas.

El problema reside en que obtener una visualización de la calidad que da por supuesta hoy en día el público cinematográfico requiere una variedad fantásticamente extensa de hardware. Una película de ciencia ficción, *El guerrero del espacio* (*Starfighter*), confió en un equipo informático para conseguir 27 minutos de sorprendentes imágenes. Pero para ello se necesitó un Cray X-MP de 12 millones de dólares conectado a dos ordenadores de un millón de dólares. Este enorme gasto se consideró que valía la pena, ya que permitió crear imágenes que habrían sido difíciles de conseguir en el mundo real.

Toda esta tecnología está muy bien para la gente que tiene acceso al equipo de grabación cinematográfico o de vídeo, a potentes ordenadores y que disponen de tiempo de sobra. Pero la mayor parte de gente que no dispone de ninguna de estas cosas también puede encontrar de gran utilidad los gráficos generados por ordenador. Por ejemplo, las imágenes animadas son una parte importante de toda clase de paquetes de Diseño Asistido por Ordenador (CAD: *Computer-Aided-Design*), y todos los buenos juegos de acción dependen en gran medida de representaciones de pantalla atractivas y con una animación bien hecha.

La sofisticación que puede conseguirse en estas imágenes está limitada por la capacidad del ordenador que utilices. El Cray opera a 100 megaflops (100 millones de operaciones de coma flotante por segundo). Pero como las imágenes deben intercambiarse varias veces por segundo para obtener una animación realista, incluso el Cray no puede generar imá-

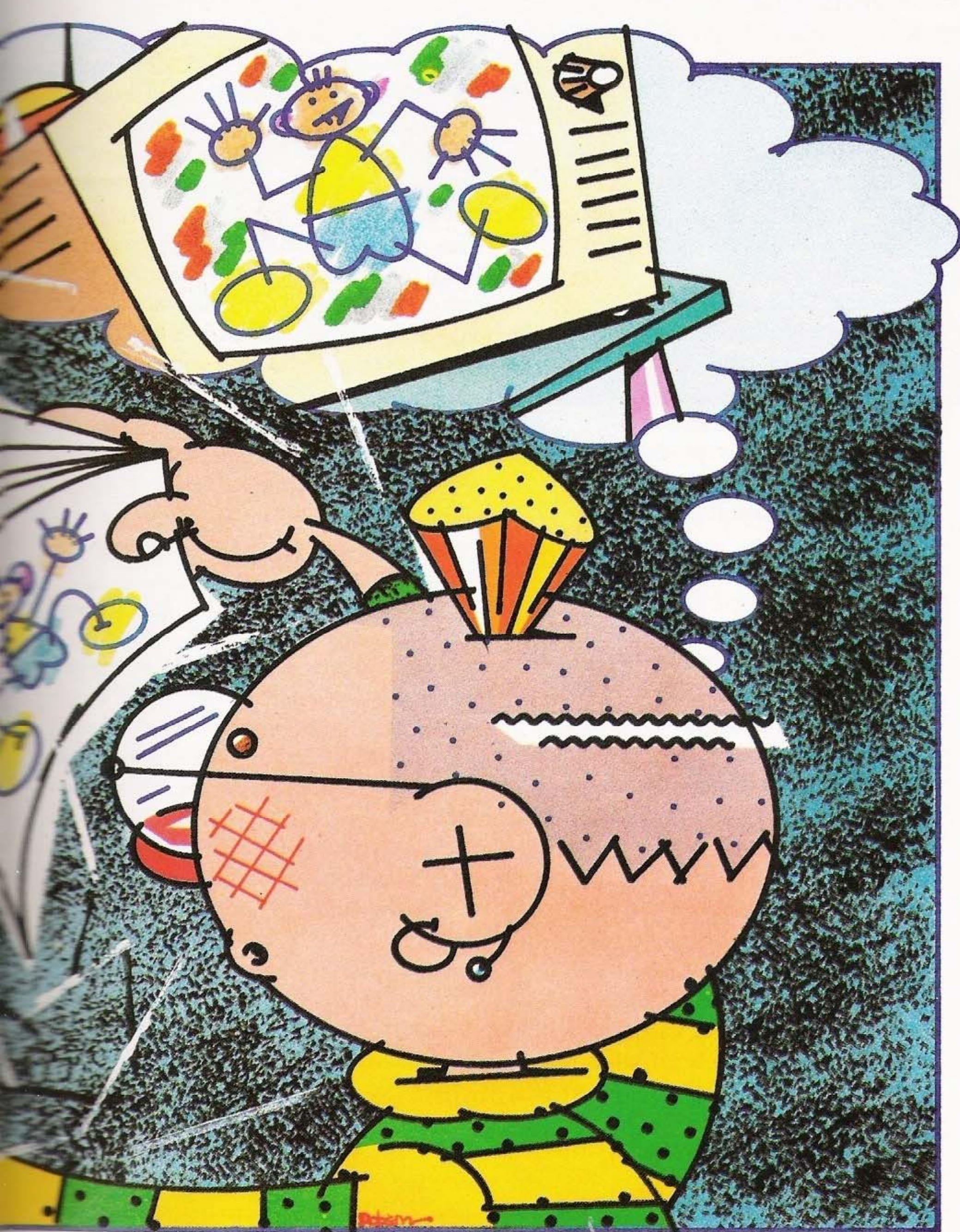


- ANIMACION POR CUADROS
- GRAFICOS POR ORDENADOR
- EXPLICACION DE LOS GRAFICOS PAGINADOS
- UN CUBO MOVIL
- CREACION DE GRAFICOS
- ANIMACION EN TIEMPO REAL
- PERSISTENCIA DE LA VISION
- ANIMACION DE PELICULAS COMERCIALES

genes de calidad lo bastante rápidamente como para conseguir una animación en tiempo real. En vez de ello, las imágenes generadas se filman por separado según un proceso de cuadros convencional.

Los niveles de detalle inferiores de las imágenes permiten que los cuadros se generen más rápidamente. Realmente, existe un simulador de vuelo en el cual las representaciones razonablemente exactas de un avión en vuelo son generadas a razón de 50 veces por segundo, obteniendo efectos animados de gran realismo.

La razón de la dificultad de la animación por ordenador a gran velocidad reside en la cantidad de información necesaria para cada imagen. Cuanto más detallada sea ésta, más memoria se necesita para almacenarla. Incrementa el número de colores disponibles, y la cantidad de RAM requerida para almacenar la información relativa al color crecerá también.



Cuanta más memoria se utilice para almacenar la imagen, más trabajo tendrá que llevar a cargo la CPU para actualizarla. La razón por la cual la mayor parte de películas comerciales generadas por ordenador emplean la animación por cuadros es simplemente que el procesador no es lo bastante potente como para actualizar grandes áreas de memoria rápidamente. Si la actualización de los gráficos es un proceso lento, incluso para quienes utilizan ordenadores potentes, ¿cómo puede un programador casero obtener animación utilizando un micro? Una solución consiste en emplear *gráficos paginados*.

¿QUE SON LOS GRAFICOS PAGINADOS?

Todos los ordenadores domésticos disponen de un área de memoria asociada a la pantalla. Puede tratarse de *memoria mapeada*, lo que significa que a cada posición de pantalla le corresponde una posición de memoria, o puede estar organizada utilizando un fichero de visualización.

Con los gráficos paginados, en vez de construir la siguiente imagen directamente en la memoria de pantalla, se reserva un área en alguna otra parte para dicho propósito. Una vez completa la nueva imagen, se copia a la RAM normalmente asociada a la pantalla. Estas áreas de pantalla adicionales se denominan páginas, por lo cual se conoce a esta técnica como gráficos paginados.

Obviamente, si sólo deseas dibujar una imagen, los gráficos paginados pueden parecer de poca utilidad. Sin embargo, si deseas escribir un programa en el cual una página de texto vaya seguida de una imagen, piensa cuán conveniente sería que fueras capaz de empezar creando la primera visualización gráfica en algún otro lugar de la memoria mientras el usuario está ocupado leyendo toda la página de instrucciones. Cuando sea necesario el gráfico, se ahorra tiempo, ya que éste se encuentra presente en otra área de la RAM. Si es conveniente, puedes emplear todo el tiempo en hacer cálculos por ti mismo y emplear el ordenador para visualizar las imágenes tan rápidamente como sea necesario.

Pero la ventaja real de los gráficos paginados se consigue cuando deseas visualizar más de una imagen en rápida sucesión. Los comandos gráficos del BASIC habitualmente escriben solamente en la memoria de pantalla. Esto significa que las imágenes se construyen en la pantalla y a continuación se salvan. Mientras que la utilización de gráficos paginados no aporta ninguna rapidez a la construcción física de una imagen en la pantalla, una vez efectuados los cálculos, dichas imágenes pueden transportarse de la memoria a la pantalla muy rápidamente. Así, los gráficos paginados conservan la simplicidad del BASIC, combinada con una velocidad de representación mucho mayor. Y si dispones varias imágenes en distintas áreas de memoria, puedes llamarlas en una secuencia rápida para permitir representaciones bastante complejas.

Supongamos que deseas construir una secuencia animada simple que muestre la rotación de un cubo. Has decidido que cuatro imágenes serán suficientes para representar una rotación del cubo y deseas que éste gire cinco veces. La estructura de un programa típico podría parecerse a esto:

```
for contador = 1 to 5 do
begin
limpiar la pantalla
presentar imagen número 1
limpiar la pantalla
presentar imagen número 2
limpiar la pantalla
presentar imagen número 3
limpiar la pantalla
presentar imagen número 4
end
```

La idea es bastante simple. Se limpia la pantalla, cada una de las cuatro imágenes se dibuja por turno y se repite el proceso hasta que el cubo haya girado cinco veces. La desventaja de este método es que dibujará cada imagen cinco veces. Los cálculos durarán un cierto tiempo cada vez, por lo que la animación resultante será pobre, con un «salto» apreciable entre cada imagen.

Veamos ahora un algoritmo alternativo, que muestra el procedimiento general de un programa estructurado en torno a las técnicas de gráficos paginados:

```
limpiar pantalla
formar imagen número 1
almacenar datos de pantalla en página de memoria 1
limpiar la pantalla
formar imagen número 2
almacenar datos de pantalla en página de memoria 2
limpiar la pantalla
formar imagen número 3
almacenar datos de pantalla en página de memoria 3
limpiar la pantalla
formar imagen número 4
almacenar datos de pantalla en página de memoria 4
for contador = 1 to 5 do
presentar memoria de página 1 a pantalla
presentar memoria de página 2 a pantalla
presentar memoria de página 3 a pantalla
presentar memoria de página 4 a pantalla
end
```

El programa es más largo y todavía tienes que pasar por el lento proceso de calcular y dibujar las cuatro imágenes, pero la animación no empieza hasta que se ha realizado dicho proceso. Una vez almacenadas en memoria, las imágenes pueden volver a llamarse con extraordinaria rapidez.

Aunque el dibujo se efectúa todavía en BASIC, el trozo de programa real que mueve una imagen determinada en memoria será una corta rutina en código máquina. Esta transfiere una pantalla entera de información más deprisa de lo que el ojo puede ver, la esencia de la animación.

DEMOSTRACION DE GRAFICOS

El siguiente programa demuestra una aplicación simple de los gráficos paginados. Como el programa contiene código máquina para obtener la necesaria velocidad en el intercambio de las imágenes, sálvalo antes de ejecutarlo para evitar percances.

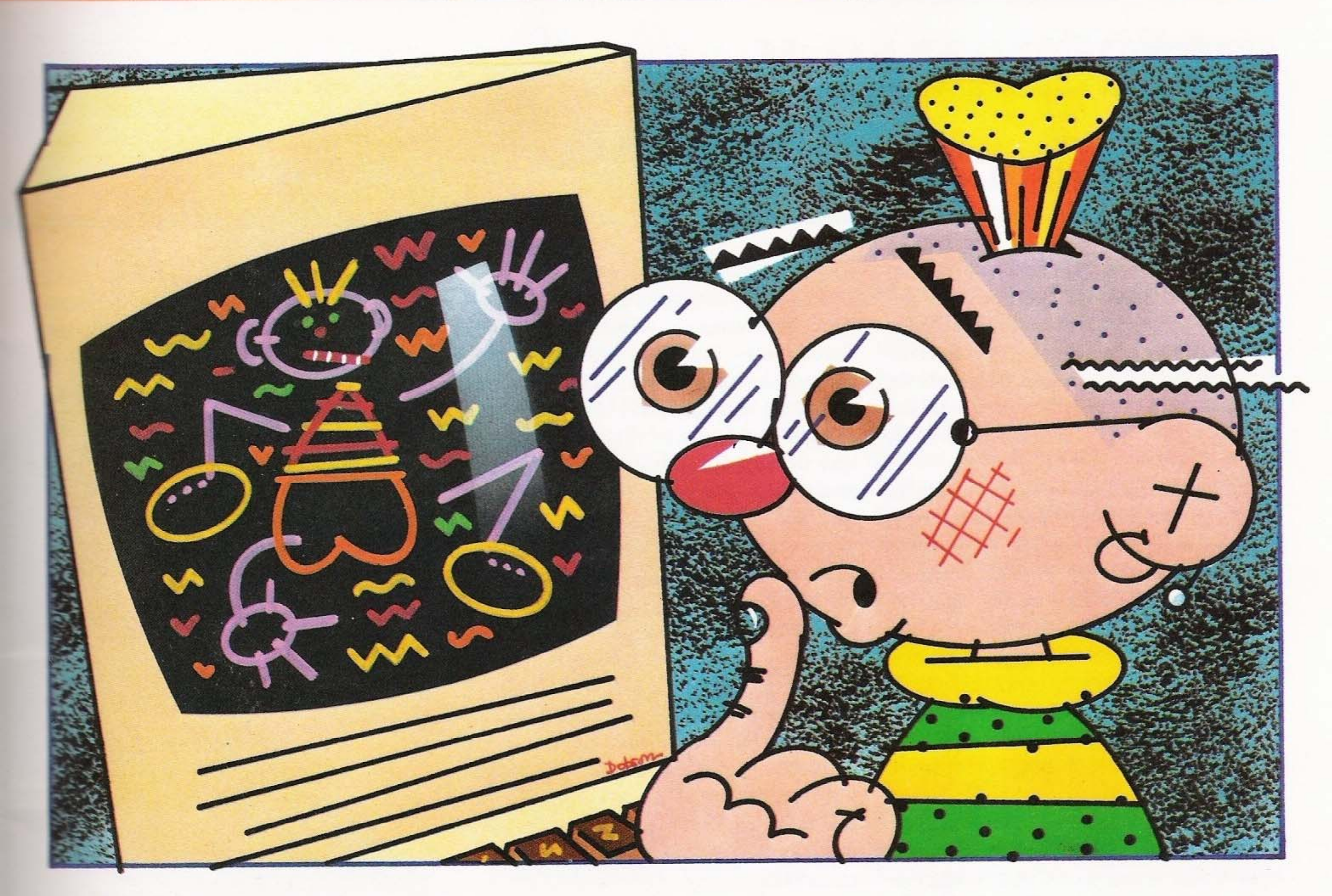
Se te pide que pulses una tecla después de haberse dibujado la imagen y debes pulsar otra tecla más para empezar la alternancia de las imágenes.

Puedes utilizar este programa como base para tus propios experimentos, cambiando las imágenes dibujadas. Un próximo artículo explicará con más detalle las técnicas utilizadas, así como el modo de llevar la sofisticación hasta los límites de tu ordenador.

La utilización de los gráficos de excelente resolución del Commodore, accesibles utilizando el cartucho Simons' BASIC o la facilidad de gráficos en alta resolución que INPUT te explicó en otros números, ofrece un inusual e interesante par de imágenes alternativas.

Las dos imágenes son similares y por ello no se dibujan individualmente. En vez de ello, se construyen utilizando un bucle FOR...NEXT que envía al programa dos veces a la rutina de dibujo, efectuando la segunda vez pequeños cambios de las instrucciones.

Si se utiliza el cartucho Simons' BASIC, el programa es correcto. Si se utiliza la facilidad de gráficos en alta resolución de INPUT tal como se publicó en INPUT anteriores, borra la línea 65, modifica el 224 en negrita de la línea 240 a 32 y el 255 en negrita de la línea 250 a 63. Cuando se publique el resto de la facilidad de alta resolución de INPUT, la línea 65 podrá dejarse en su sitio, pero los cambios efectuados en las líneas 240 y 250 deberán dejarse como permanentes.



Además, debes preceder todos los comandos gráficos con una @.

```
20 POKE 51,255:POKE 52,
   29:POKE 55,255:POKE 56,
   29:CLR
30 GOSUB 220
40 D=64
50 FOR N=0 TO 1
60 HIRES 0,1:MULTI 2,4,
   5:COLOUR 6,6
65 CIRCLE 80,100,30+10*N,
   30+10*N,1:PAINT 80,100,2
70 FOR X=0 TO 159:PLOT
   RND(1)*160,RND(1)*200,
   RND(1)*4
80 PLOT X,100+30*SIN((N
   +X/50)*EXEC /4),3
85 LINE X,50+N*50,0,100-
   N*50,RND(1)*4+1
90 NEXT X
100 GOSUB 430:IF N=0
    THEND=96
110 TEXT 30,1,"PRESS ANY KEY",
    3,1,8:POKE 198,0:WAIT198,1
120 NEXT N
150 D=64:FOR N=0 TO 1
170 GOSUB 440:IF N=0
    THEND=96
190 NEXT N
200 GOTO 150
220 FOR Z=7680 TO 7738:READ
    X:POKE Z,X:NEXT Z:RETURN
230 DATA 169,0,141,14,220,
    169,53,133,1
240 DATA 169,0,133,251,133,
    253,169,224,133,252,169,
    64,133,254,160,0
250 DATA 177,251,145,253,
    192,63,208,16,165,252,
    201,255,208,10
260 DATA 162,1,142,14,220,
    162,55,134,1,96,200
270 DATA 208,229,230,252,
    230,254,76,25,30
430 POKE 7700,D:POKE 7706,
    251:POKE 7708,253:SYS
    7680:RETURN
440 7700,D:POKE 7706,
    253:POKE 7708,251:SYS
    7680:RETURN
```

La línea 20 del programa limpia un espacio de memoria para el código máquina y la pantalla que necesitamos almacenar. La línea 30 envía a continuación el programa a la subrutina de las líneas 220 a 270, en las que los datos se colocan en memoria. La línea 40 determina la posición de memoria en donde se va a almacenar la primera de las dos pantallas y la 50 en un bucle FOR...NEXT para las distintas pantallas. El modo gráfico y el color de la pantalla (azul) se determinan en la línea 60.

Los dibujos primero y segundo se dibujan en las líneas 65 a 85. En la línea 100, el programa se envía primeramente a la línea 430, que salva la pantalla antes de cambiarla de posición. Se intercambian las dos pantallas por medio de una rutina contenida en las líneas 150 y 200.

HISTOGRAMAS Y TARTAS ESTADISTICAS

Sea cual sea la procedencia de tus datos (un presupuesto familiar, un pequeño negocio, una afición o los mismos indicadores de tu salud), notarás la diferencia si los representas mediante un diagrama de barras o una tarta redonda.

Ya has podido ver en nuestra revista cómo se escribe un programa que visualiza tus datos en forma gráfica. Otra manera de representar la información numérica sería el diagrama no lineal. Las barras (histogramas) y las «tartas» están entre las representaciones más ampliamente utilizadas de la información comercial y estadística. Y es que ofrecen una ventaja adicional: pueden colorearse y adquirir formas deslumbrantes.

Aparte de esto, cada diagrama se adapta particularmente mejor a un tipo de datos que a otro. Las barras se prestan a ser utilizadas para datos que fluctúan a lo largo de un amplio abanico de valores. El diagrama en forma de tarta está indicado en aquellas ocasiones en que te interese mostrar los distintos valores en proporción a la totalidad: por ejemplo, si tienes que comparar porcentajes.

Los programas que siguen ilustran la manera de instruir a tu ordenador para que te haga ambos tipos de diagramas.

Ya sabes que tu Commodore funcionará mejor (dado que sus funciones de gráficos son poco accesibles en el BASIC normal) con el cartucho BASIC de Simon.

LA RUTINA DE LAS BARRAS

Esta rutina es en esencia la misma que la de cualquier gráfico. Una vez reunidos los datos, hay que entrarlos en la memoria, determinar las coordenadas, trazar las barras y añadir los textos. Una de dos: o introduces (INPUT) cada uno de los datos y vas trazando cada barra sobre la marcha, o esperas a tener introducidos todos los datos para empezar la tarea del trazado de barras. Existe una tercera posibilidad: leer (READ) los datos y trazar las barras, cambiándolos siempre que desees dibujar un diagrama diferente. Lo que sí te aconsejamos es que, sea el que sea el método a seguir, almacenes los datos en una tabla de variables, porque así la máquina identificará fácilmente las coordenadas de cada barra.

LEER LOS DATOS

Entra estas líneas que leen (READ) los datos, y ejecútalas (RUN). Todavía no verás nada en pantalla, pero harás bien en ejecutar cada parte del programa, para comprobar y depurar errores.

```
5 HIRES Ø,:MULTI 2,4,6
1Ø N=12
2Ø DIM A(N)
7Ø FOR T=1 TO N
8Ø READ A(T)
9Ø NEXT T
3Ø1Ø DATA 2,4,7,4,6,3,8,Ø,5,
      6,9,4
```

Esta sección del programa selecciona un modo que sirva para gráficos en aquellas máquinas que lo requieren. Establece el 12 como número de barras que se trazarán (línea 10) y dimensiona la tabla con este mismo número (línea 20). Escoge, si quieres, otro número, pero si es uno mayor, no olvides que has de tener un número equivalente de entradas en la sentencia DATA de la línea 3010, porque de lo contrario te encontrarás con el error *out of data*. Con frecuencia es mejor disponer de más entradas en la línea de DATA cuando se está comprobando el programa, de este modo puedes aumentar o disminuir el número de barras a trazar en la línea 10, sin que sea necesario alterar los datos cada vez.



LAS ESCALAS

Ya tiene el ordenador en su poder los valores absolutos de cada coordenada de barras, pero habrás de decirle a qué escala los quieres representados. Si no se lo dices, puede que alguno de los datos te rebase la pantalla, mientras que los datos más pequeños aparezcan en ella como motitas prácticamente invisibles. Para trazar los ejes a escala, escribe y ejecuta (RUN) las si-

- COMO SE DIBUJAN LAS BARRAS
- TOMA DE DATOS
- LA ESCALA DE LAS COORDENADAS
- ESTADISTICA EN 3-D
- EL DIAGRAMA DE TARTA



guientes líneas (de nuevo, hazlo como prueba, aunque nada veas todavía en la pantalla).

```
30 DX=150/N
40 READ DY
3000 DATA 18
```

Esta parte del programa produce una escala para el eje X dividiendo el área disponible de la pantalla (descontados los márgenes) por el número de barras que se van a trazar (línea 30). El número máximo de barras posible depende de cada ordenador, pero puedes deducir el valor que mejor gráfico te ofrece si conoces la extensión de la pantalla de gráficos de tu ordenador (la encontrarás en el Manual del Usuario): cambia en consonancia el valor de N en la línea 30.

La escala para los valores del eje Y se obtiene multiplicándolos por un factor, teniendo en cuenta el espacio que necesitas para el texto que acompañará el gráfico. Este valor se entra automáticamente, pero es leído, READ (línea 40), de la sentencia DATA (línea 3000). En este punto, puede que te tiente escribir una breve rutina para entrar, INPUT, este valor en la línea 40. Si sabes cómo hacerlo, borra la línea 3000; de otra manera, ésta será el primer tramo de datos que será dibujado, en lugar del primer número contenido en la línea 3010.

El resto del programa comprende dos rutinas, una para dibujar los ejes y otra para dibujar las barras. Escribe esta parte del programa, pero no la ejecutes (RUN), porque recibirás un mensaje de error si el ordenador no encuentra las rutinas que han sido llamadas:

```
100 GOSUB 1000
110 FOR T=1 TO N
130 X=(T-1)*DX+11
140 GOSUB 2000
150 NEXT T
160 GOTO 160
```

La línea 100 bifurca el programa a la rutina que dibuja los ejes, y la línea 110 agota el número de barras que han de dibujarse. Por cada vuelta del bu-

cle, la línea 130 da escala a la coordenada X correspondiente a la barra que se va a trazar (*DX) y añade un *offset* (un 11 en el Commodore 64) para dejar un margen en el eje Y. La línea 140 lleva el programa a la subrutina que dibuja las barras, y la línea 150 pasa al siguiente incremento del contador del bucle.

APARECEN LAS BARRAS

Entra ahora las rutinas que dibujan los ejes y trazan las barras:

```
1ØØØ LINE 1Ø,Ø,1Ø,181,3:LINE
     1Ø,181,16Ø,181,
     3
1Ø5Ø FOR T=Ø TO 1Ø
1Ø6Ø TEXT Ø,18Ø-T*18,
     MID$(STR$(T),2),1,
     12
1Ø7Ø NEXT T
1Ø8Ø RETURN
2ØØØ CO=CO+1:IF CO>3
     THENCO=1
2Ø1Ø BLOCKX,180-A(T)*DY,
     X+DX-1,18Ø,
     CO
2Ø6Ø RETURN
```

Para dibujar los ejes, esta parte del programa mueve el cursor a la izquierda de la pantalla arriba, reservando margen y la línea 1000 dibuja el eje Y.

La misma línea dibuja el eje X a partir del cabo inferior del eje Y y hacia la derecha de la pantalla.

Las líneas de la 1050 y a la 1070 establecen un bucle con números del 0 al 10 y los imprime a lo largo del eje Y. El resto del programa establece colores alternantes (línea 200) para las barras.

La línea 2010 las dibuja.

TRES DIMENSIONES

Cuando ejecutes este programa, notarás que el diagrama de barras resulta de una apariencia muy poco escolar pero que gana en atractivo a cualquier representación lineal, a pesar de que ambos se basen en los mismos datos. Pues bien, todavía se puede realzar la visualización de la información presentándola en tres dimensiones. Esto posibilita un buen uso del color y una figura sólida y natural.

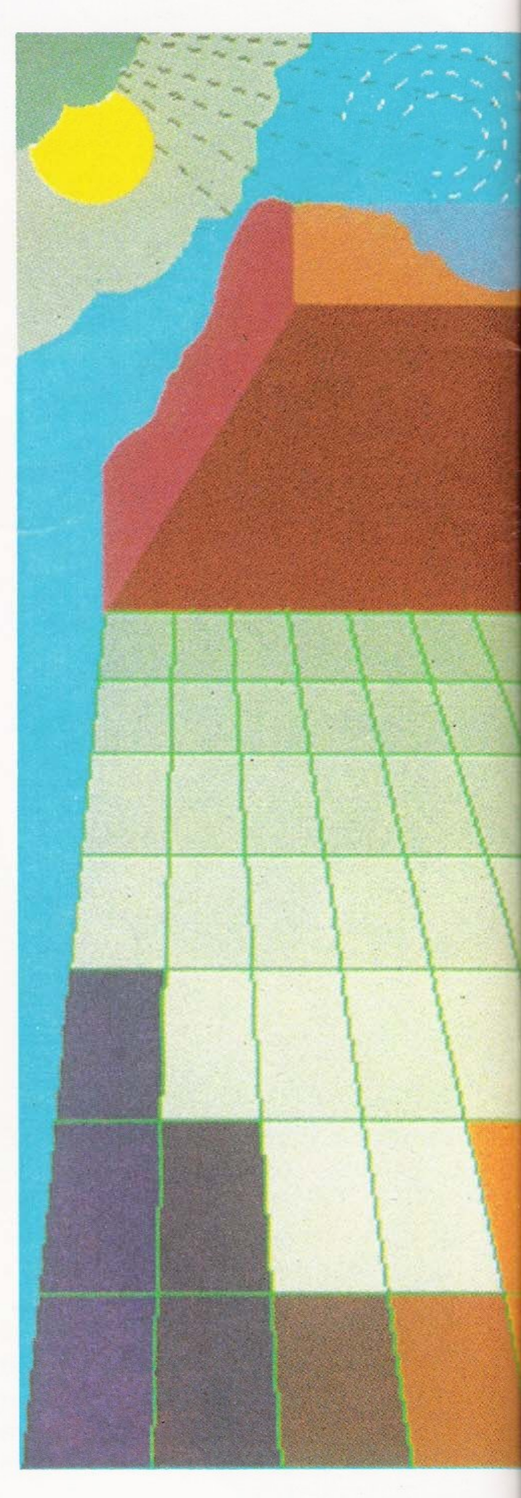
Antes de desarrollar el siguiente programa, guarda lo que ya has escrito (para saber lo que tienes) y, sin hacer un NEW o un BREAK, introduce los siguientes cambios. Debes saber editar las facilidades de tu ordenador para hacer los cambios y ahorrarte esfuerzos, pero sobre todo asegúrate de que no cometes errores dejando pasar pequeñas diferencias en las líneas del listado.

```
2Ø1Ø BLOCKX,18Ø-A(T)*DY,
     X+(DX*.5)-1,18Ø,CO
```

No ejecutes el programa ahora, tal como está, puesto que no está completo y lo único que vas a conseguir es un mensaje de error. Las alteraciones que acabas de hacer establecen algunas variables y los comandos adicionales para dibujar en la dirección Z, la tercera dimensión. Para completar el programa entra las siguientes líneas y ejecútalo (RUN).

```
1Ø1Ø LINE 1Ø+DX*.5,Ø,
     1Ø+DX*.5,181,
     3
1Ø2Ø FOR Z=18 TO 18Ø
     STEP18
1Ø3Ø LINE 1Ø,Z,1Ø+DX*.5,Z-5,
     1
1Ø4Ø NEXT
1Ø5Ø FOR T=Ø TO 1Ø
2Ø2Ø FOR N=Ø TO DX*.
     5-1
2Ø3Ø LINE X+N,18Ø-A(T)*DY-1,
     N+X+(DX*.5)-1,18Ø-
     A(T)*DY-5,1
2Ø4Ø NEXT
2Ø5Ø LINE X+(DX*.5)-1+N,18Ø-
     A(T)*DY-5,X+(DX*.5)-
     1+N,18Ø,CO
```

Ahora ya debes tener sobre tu pantalla una espléndida representación en tres dimensiones de tus datos. Es un útil ejercicio cambiar los valores de los comandos del color para seleccionar aquellos que a ti más te gustan. Anota seguidamente aquellas líneas que habrás de cambiar para alterar la escala del gráfico, de modo que las tengas siempre a mano cuando vayas a utilizar el programa en un futuro. La forma más sencilla de hacer notas es mediante sentencias REM. Por ejemplo, puedes añadir una nueva línea que contenga una sentencia REM seguida de una línea alternativa, que incluye su número y el comando correspondiente.



TARTAS

Una manera igualmente atractiva de representar gráficamente la información es el diagrama de tarta. Se trata de un gráfico circular donde se traza una única coordenada (polar) para cada tipo de datos. El tamaño de cada división está representado por un ángulo.

En su estructura, el programa que dibuja una tarta es de lo más sencillo, pero nosotros lo hemos complicado un poco para que lo puedas utilizar con facilidad.

Entralo y ejecútalo (RUN) si quieres ver cómo funciona:

```
1Ø DIM A(1ØØ),P(1ØØ)
2Ø PRINT "[SHIFT+CLR/HOME
   ][CTRL+8]":COLOUR Ø,Ø
4Ø PRINT AT(17,1Ø)"MENU"
4Ø PRINT
   TAB(1Ø)"[CTRL+2][CRSR
   ABAJO]1: ENTRADA DE
   DATOS"
6Ø PRINT TAB(1Ø)"2: VER
   DIAGRAMA"
7Ø PRINT TAB(1Ø)"3: FIN DE
   PROGRAMA"
8Ø GET G$:G=VAL(G$):IF G<1
   OR G>3 THEN8Ø
9Ø ON G GOSUB 2ØØ,4ØØ,
   6ØØ
1ØØ GOTO 2Ø
2ØØ PRINT
    "[SHIFT+CLR/HOME]"
    :N=Ø
21Ø A$"":PRINT "VALOR N.",
    N+1,:INPUT A$:IF A$=""
    OR VAL(A$)=Ø
    THENRETURN
22Ø N=N+1:A(N)=
    VAL(A$)
23Ø IF N<31 THEN
    21Ø
24Ø RETURN
4ØØ IF N=Ø THEN
    RETURN
4Ø5 PRINT "[SHIFT+CLR/HOM
    E]":COLOUR 1,1:HIRESØ,
    1:MULTI2,4,6
41Ø TT=Ø:FOR T=1 TO
    N:TT=TT+A(T):
    NEXT
42Ø RT=Ø:FOR T=1 TO N:RT
    =RT+A(T):P(T)=RT/TT:NE
    XT T
43Ø CO=1:P(Ø)=-1:
    N=Ø
45Ø FOR T=Ø TO 2*EXEC
    STEP.Ø1
46Ø IF T>2*EXEC *P(N)
    THENN=N+1:CO=CO+1:IF
    CO>3 THENCO
    =1
47Ø LINE 8Ø,1ØØ,
    8Ø+4Ø*SIN(T),
    1ØØ+5Ø*COST(T),
    CO
48Ø NEXT T:LINE 8Ø,1ØØ,8Ø,
    15Ø,Ø
49Ø PAUSE 15:NRM :
    RETURN
6ØØ PRINT "[SHIFT+CLR/HOM
    E][CTRL+1]":NRM :COLO
    UR 6,1
```

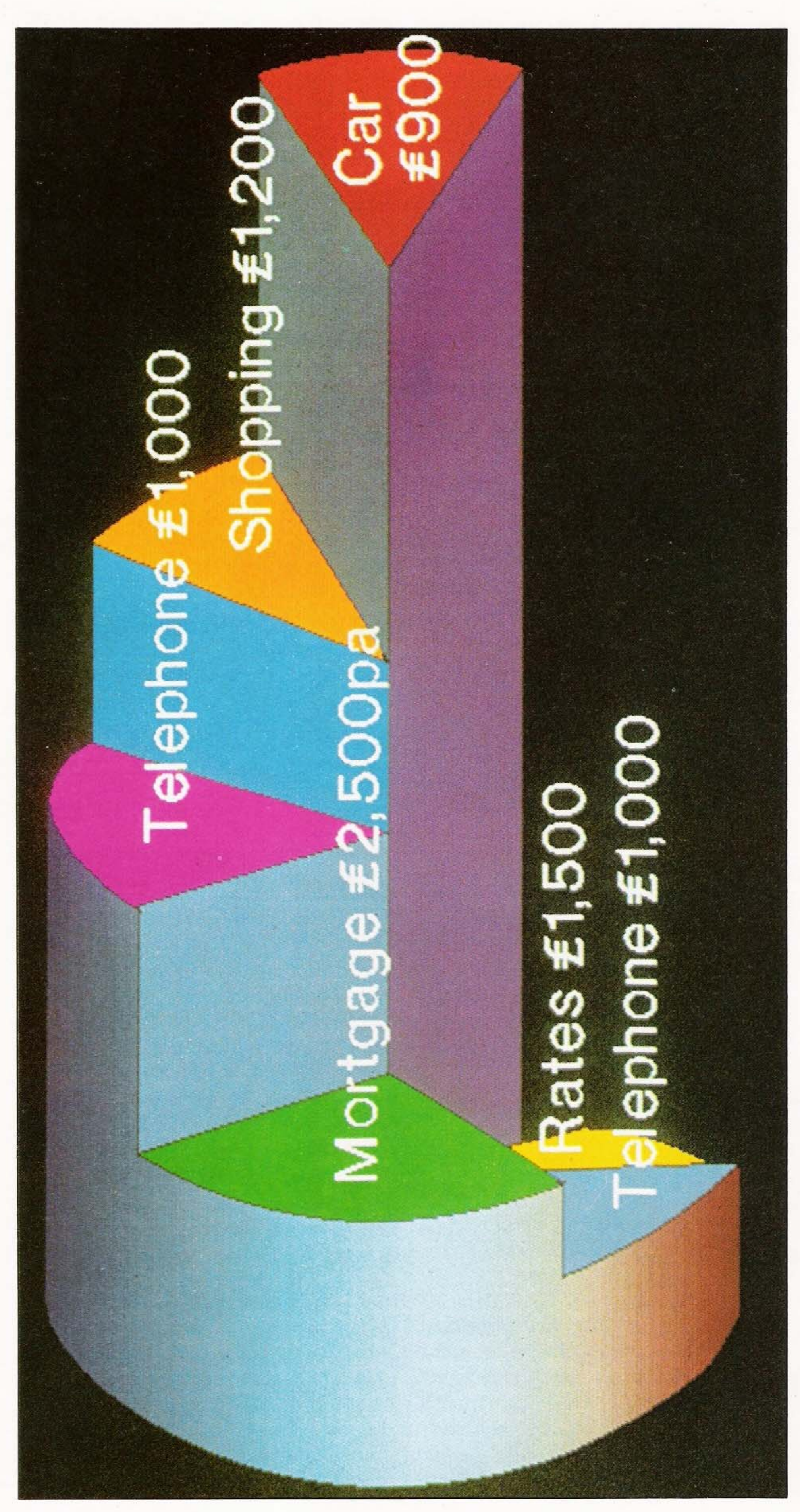
DESCRIPCION DEL PROGRAMA

Cuando hayas ejecutado este programa, verás primero un menú. Es lo que imprime (PRINT) las líneas 40 a 70 y te permite entrar los datos, ver el diagrama o finalizar el programa. La línea 80 espera a que aprietes una tecla; puede ser cualquiera (salvo las que restauran la memoria), aunque sólo 1, 2 y 3 hacen desaparecer el menú. Esto es lo que consigue el IF ... THEN establecido en la línea 80. Una condición semejante situada en la línea 400 hace que el programa vuelva al menú si aprietas 2 sin entrar ningún dato.

Y si pruebas a pulsar 3 finalizas el programa.

Si quieres que trabaje de nuevo, habrás de recurrir al RUN.

Para empezar, lo más obvio es que elijas 1, que lleva el programa hasta una rutina de entrada de datos. La línea 90 calcula la línea actual a la que



debe dirigirse el programa, la cual en este caso es la línea 200. Esta establece la variable (N) de los datos, que vamos entrando (INPUT) en la línea 210 y almacenándolos mediante la 220 en una tabla que quedó dimensionada en la línea 10.

Para entrar los datos, escribe los números seguidos de RETURN. La línea 230 comprueba si has introducido todos los datos cuyo espacio fue reservado en la línea 10. Si no es así, el programa hace un bucle y vuelve a la línea 210 para entrar el siguiente dato. Observa que no necesitas llenar todo el espacio reservado; si, por ejemplo, sólo deseas una tarta dividida en cinco porciones, habrás de pulsar RETURN dos veces una vez introducido el quinto tipo de datos. La línea 100 lleva, entonces, el programa a la rutina que visualiza el menú. Ahora bien, si tus datos llenan completamente el espacio reservado, la línea 240 enviará automáticamente el programa al menú, por lo que no podrás nunca rebasar la capacidad de la tabla.

AQUI TIENES LA TARTA

Si ahora pulsas 2, el programa va a la línea 400, que inicia la rutina que dibuja la tarta. La línea 410 hace un bucle para los datos y sus totales. El resto de la rutina determina la escala para éstos y dibuja el gráfico. Repasa, si quieres enterarte bien, lo que ya se ha publicado en INPUT sobre el modo de tratar las funciones circulares.

La línea 420 hace un bucle para los datos entrados, obteniendo el subtotal de éstos en cada pasada y divide el subtotal por el total de los datos. Estos subtotales ya a escala se almacenan en la tabla P () dimensionada en la línea 10. La línea 430 restaura algunas variables que ayudan la secuencia de colores de las porciones de la tarta. Las líneas 450 a 480 dividen la tarta, seleccionan el color de cada sección y dibujan los radios a colores para rellenar cada sección. La línea 490 establece un breve retardo para que el gráfico se haga visible, cambia después el modo y vuelve al menú. La línea 600 es una rutina aparte, que pone la visualización en normal.

PROGRAMA MULTIGESTION (y III)

A continuación publicamos el resto del programa Multigestión que, por razones de espacio, no se publicó completo en el número anterior de INPUT.

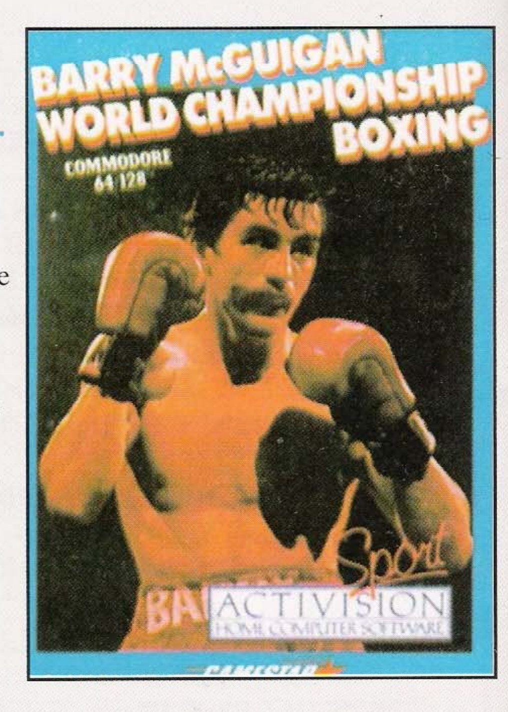
```
1515 NEXT Y
152Ø NEXT X
1525 CLOSE 1:RETURN
153Ø REM ***** 'LOAD' PROGRAMAS *****
1535 OPEN 1,1,Ø,"FECOM C64"
154Ø FOR F=Ø TO 1ØØ
1545 FOR E=Ø TO 8
155Ø INPUT#1,C$(F,E):IF C$(F,E)="-" THENC$(F,E,)=" "
1555 NEXT E
156Ø NEXT F
1565 FOR A=Ø TO 1ØØ
157Ø FOR B=Ø TO 4
1575 INPUT#1,S$(A,B):IF S$(A,B)="-" THENS$(A,B)" "
158Ø NEXT B
1585 NEXT A
159Ø FOR X=Ø TO 1ØØ
1595 FOR Y=Ø TO 1
16ØØ INPUT#1,CO:FOR Z=1 TO CO:FOR ZZ=1 TO 4:INPUT#1,D$(ZZ,Z)
16Ø5 NEXT ZZ,Z
161Ø INPUT#1,T$(X,Y):IF T$(X,Y)="-" THENT$(X,Y)=" "
1615 NEXT Y
162Ø NEXT X
1625 CLOSE 1:RETURN
163Ø PRINT "[SHIFT+CLR/HOME]":POKE 53272,21:POKE 5328Ø,Ø:POKE 53281,Ø
1635 PRINT "[SHIFT+CLR/HOME][3*CRSRABAJO]"TAB(1Ø)"[CTRL+9]*****[2ESPACIOS]MENU[
     2ESPACIOS]*****":PRINTTAB(1Ø)"[CTRL+6][2*CRSRABAJO]1.ENTRARDATOS"
164Ø PRINT TAB(1Ø)"[CRSR ABAJO]2.VER DATOS":PRINT TAB(1Ø)"[CRSR ABAJO]3.GUARDAR
     EN CINTA"
1645 PRINT TAB(1Ø)"[CRSR ABAJO]4.CARGAR DESDE CINTA":PRINT TAB(1Ø)"[CRSR ABAJO] 5
     OPCION IMPRESORA"
165Ø PRINT TAB(1Ø)"[CRSR ABAJO]6.ALTERAR DATOS":PRINT TAB(1Ø)"[CRSR ABAJO]7. MENU
     PRINCIPAL"
1655 PRINT TAB(1Ø)"[ 2*CRSR ABAJO][CTRL+9] ESCOGE LA OPCION "
166Ø GET K$:IF VAL (K$)<1 OR VAL (K$)>7 THEN166Ø
1665 C$=" ":KK$=K$:IF K$="1" THENGOTO 186Ø
167Ø IF K$="2" THENGOSUB 1825:GOTO 1935
1675 IF K$="3" THENGOSUB 144Ø
168Ø IF K$="4" THENGOSUB 153Ø
1685 IF K$="5" THENPRINT "[SHIFT+CLR/HOME]":GOSUB 1915
169Ø IF K$="6" AND CO<>Ø THENC$="S":CQ=1:QQ$=D$(4,1):GOTO 17Ø5
1695 IF K$="7" THEN555
17ØØ GOTO 1635
17Ø5 CC=Ø:C1=Ø
171Ø PRINT "[SHIFT+CLR/HOME][ 2*CRSR ABAJO]":IF C$="S" THENPRINT
     TAB(11)"[CTRL+9]ENTRADA NUMERO"CQ
```

```
1715 PRINT "[CRSRABAJO]"TAB(17-(LEN(A$(VAL(QQ$)))*.5))A$(VAL(QQ$))
172Ø PRINT"[CRSRABAJO][CTRL+6]——————————————————————";
1725 PRINT"[2ESPACIOS]FECHA[2ESPACIOS][SHIFT+B][3ESPACIOS]CONCEPTO[3ESPACIOS]
     [SHIFT+B][3ESPACIOS]CANTIDAD"
173Ø PRINT"——————————————————————":SC=Ø
1735 IF C$="S" THENC1=CQ:GOSUB 179Ø:GOTO 2Ø2Ø
174Ø C1=C1+1:IF D$(A,C1)=QQ$ THENGOSUB 179Ø:IF PR$="N" THENSC=SC+1
1745 IF SC=>1Ø THENSC=Ø:GOSUB 2ØØ5:PRINT "[SHIFT+CLR/HOME] [ 5*CRSR ABAJO]"
175Ø IF C1=4ØØ OR VAL(D$(4,C1))=Ø THEN176Ø
1755 GOTO 1735
176Ø C1=Ø:FOR Z=1 TO 8:N(Z)=Ø:NEXT :FR=Ø
1765 C1=C1+1:IF C1=>4ØØ OR VAL (D$(4,C1))=Ø THENRETURN
177Ø IFVAL(D$(4,C1))=VAL(QQ$)THENN(VAL(QQ$))=N(VAL(QQ$))+VAL(D$(3,C1))
1775 IFVAL(D$(4,C1))=8ANDVAL(QQ$)=8THEN1765
178Ø IFVAL(D$(4,C1))<>8THENFR=FR+VAL(D$(3,C1))
1785 GOTO 1765
179Ø PRINT LEFT$ (D$ (1,C1)+"[ 19 ESPACIOS]",9)"[SHIFT+B]";
1795 PRINT LEFT$ (D$ (2,C1)+"[ 19 ESPACIOS]",14)"[SHIFT+B]";
18ØØ VV$=D$(3,C1):TA=14
18Ø5 VV=VAL(VV$):IF VV-INT (VV)=Ø THENVV$=STR$(VV)+".ØØ"
181Ø IF MID$(VV$,LEN(VV$)-1,1)="." THENVV$=VV$+"Ø"
1815 PRINT RIGHT$("[16 ESPACIOS]"+VV$,TA)
182Ø RETURN
1825 PRINT "[SHIFT+CLR/HOME][ 6*CRSR ABAJO]"
183Ø PRINTTAB(1Ø)"[CTRL+9][5ESPACIOS]CATEGORIA[6ESPACIOS][CRSRABAJO][CTRL+6]":
     POKE198,Ø
1835 FOR Z=1 TO 8:PRINT TAB(9)Z": "A$(Z):NEXT
184Ø PRINT TAB(1Ø)"[CRSR ABAJO][CTRL+9][ 2 ESPACIOS]ESCRIBE OPCION
     ?[SHIFT+ESPACIO]"
1845 GET K$:IF (VAL (K$)<1 OR VAL(K$)>8) AND K$<>CHR$(13) THEN1845
185Ø IF K$=CHR$(13) AND KK$="1" THEN1845
1855 PRINT "[SHIFT+CLR/HOME]":QQ$=K$:RETURN
186Ø IF K$=CHR$(13) THENCO=CO-1:GOTO 1635
1865 CO=CO+1:IF CO>4ØØ THENCO=4ØØ:GOTO 1635
187Ø C1=CO:D$(1,C1)="  "
1875 PRINT "[SHIFT+CLR/HOME][ 2*CRSR ABAJO]":IF C$<>"S" THENPRINT "PULSA RETURN
     EN CAMPO DE FECHA PARA MENU"
188Ø INPUT "[ 3*CRSR ABAJO][ 2 ESPACIOS]FECHA";D$(1,C1):IF D$(1,C1)="  "
     THENK$=CHR$(13):GOTO 186Ø
1885 INPUT "[CRSR ABAJO][ 2 ESPACIOS]ESCRIBE EL ARTICULO[CTRL+6]";D$(2,C1)
189Ø INPUT"[CRSRABAJO][2ESPACIOS]ESCRIBELACANTIDAD[CTRL+6]";D$(3,C1):GOSUB1825
1895 IF QQ$<>CHR$(13) THEND$(4,C1)=QQ$
19ØØ IF QQ$=CHR$(13) THEN1635
19Ø5 IF C$="S" THENQQ$=D$(4,CQ):GOTO 17Ø5
191Ø GOTO 186Ø
1915 PRINT "[SHIFT+CLR/HOME]"TAB(5)"[ 4*CRSR ABAJO]QUIERES IMPRIMIR? (S/N)"
192Ø GET K$:IF K$="S" THENPR$=S":RETURN
1925 IF K$="N" THENPR$="N":RETURN
193Ø GOTO 1915
1935 IF K$=CHR$(13) THEN1635
194Ø IF PR$="S" THENOPEN 4,4:CMD 4
```

```
1945 GOSUB 17Ø5:PRINT "————————————————————————[CTRL+6]"
195Ø VV$=STR$(N(VAL(QQ$)))
1955 PRINT TAB(18)"TOTAL :";:TA=12:GOSUB 18Ø5
196Ø PRINT "[CTRL+6]————————————————————————"
1965 PRINT TAB(11)"[CRSR ABAJO][CTRL+6]GASTOS TOTAL :";:TA=12
197Ø VV$=STR$(FR):GOSUB 18Ø5:VV$=STR$(N(VAL(QQ$))-FR)
1975 IF QQ$="8" THENPRINT TAB(16)"[CTRL+6][CRSR ABAJO]BALANCE :";:TA=12:GOSUB 18Ø5
198Ø IF PR$="S" THENPRINT#4,CHR$(13):CLOSE 4
1985 GOSUB 2Ø6Ø:K$="2":POKE 198,Ø
199Ø GET W$:IF W$="  " THEN199Ø
1995 IF W$="S" THENGOSUB 18Ø5:GOTO 1935
2ØØØ GOTO 1665
2ØØ5 PRINT TAB(11)"[ 2*CRSR ABAJO](PULSA UNA TECLA PARA CONTINUAR) [CTRL+6]":POKE
     198,Ø: WAIT198,1
2Ø1Ø PRINT "[SHIFT+CLR/HOME][ 6*CRSR ABAJO]";
2Ø15 FOR Z=1 TO 4:PRINT "[ 39 ESPACIOS]":NEXT :RETURN
2Ø2Ø PRINT"[CRSRABAJO](,)PARAIRHACIAATRAS(.)PARAADELANTE[10ESPACIOS][CRSRABAJO]
     'SPC'PARACAMBIAR"
2Ø25 GET P$:IF P$="  "THEN2Ø25
2Ø3Ø IF P$=CHR$(13) THEN1635
2Ø35 IF P$=CHR$(32) THEN1875
2Ø4Ø IF P$="." THENCQ=CQ+1:IF CQ>CO THENCQ=CO
2Ø45 IF P$="," THENCQ=CQ-1:IF CQ<1 THENCQ=1
2Ø5Ø IF P$="," OR P$="." THENQQ$=D$(4,CQ):GOTO 17Ø5
2Ø55 GOTO 2Ø25
2Ø6Ø PRINT "[CRSR ABAJO]PULSA UNA TECLA PARA SEGUIR O <RETURN>":PRINT "PARA IR AL
     MENU"
2Ø65 RETURN
```



GOLPES Y PORRAZOS

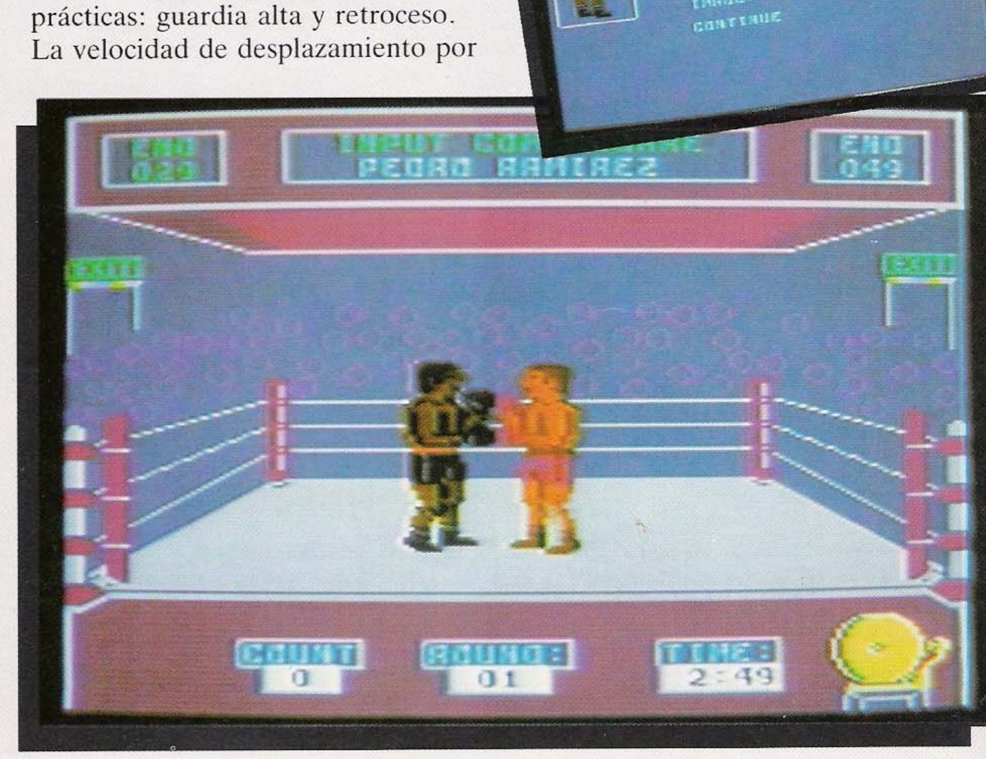


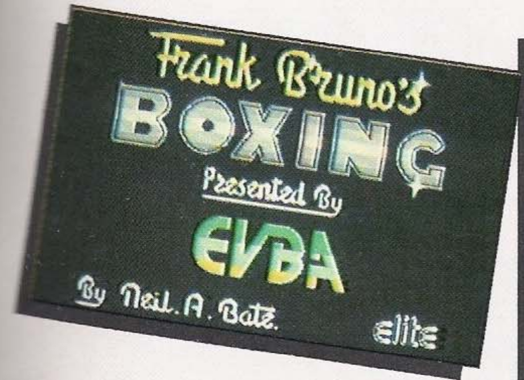
Muchos de los juegos de acción que más récords de ventas han batido se basan en este tipo de juego: tu palanca se convertirá en una especie de antebrazo que será el que dé y reciba los porrazos. El ordenador se convertirá en el instrumento que recibirá todos los golpes, y que te permitirá luchar a muerte, dando y recibiendo hasta que te duelan las manos de tanto darle marcha al joystick.
A todos aquellos que, cuando visionan una película de BRUCE LEE o de JACKIE CHANG se quedan pensativos, estos juegos les permitirán ponerse en la piel de sus héroes favoritos.
Cabe mencionar que es bastante evidente que los juegos de ARTES MARCIALES suelen ser más atractivos, ya que caben escenarios más exóticos, golpes más extravagantes y sonidos tan estremecedores como el KIAI !!! Quizás por eso suelen parecer más monótonos los juegos de lucha libre o de boxeo que las aventuras orientales con ninjas, guerreros y estrellas de la muerte.

BOXEO

BARRY MAC GUIGAN CHAMPIONSHIP

Sin duda uno de los primeros boxeos que salió al mercado fue el BARRY MAC GUIGAN. En él se perfilaban ya las características que iban a tener los sucesores: el ángulo de la pista, la elección de entre diferentes características de jugadores, la posibilidad de múltiples cambios en la vestimenta y la posibilidad de entrenamientos.
Lamentablemente, en tema de juegos el que innova un concepto, y sin duda alguna éste es el caso, paga las consecuencias con un menor grado de detalle y, a menudo, el esfuerzo creativo se ve posteriormente «recompensado» por un alto grado de plagio posterior.
En el BARRY podrás elegir de entre varios tipos de peleones: desde el tipo BULLDOG (para los más bestias) hasta el estilo bailarín para los contrincantes de tipo técnico que amen los rápidos juegos de piernas.
El juego incluye la posibilidad de entrenamiento intensivo en una sala de entrenamiento en la que podrás pasarte varias horas trabajando el saco de serrín o bien usar la cuerda para aumentar la velocidad de piernas y hacer abdominales. Tras todo esto llega la definitiva pelea: aparecerá en la parte inferior de la pantalla el GONG, el ring visto desde uno de los laterales, las puntuaciones de los diferentes contrincantes, el tiempo que queda hasta el final del juego, etc. Existen siete tipos de golpes: cinco tipos de ataques y dos tipos de defensa. Los ataques incluyen desde el uppercut hasta un velocísimo golpe de izquierda a los abdominales del contrario. Las defensas son menos variadas pero no por ello menos prácticas: guardia alta y retroceso.
La velocidad de desplazamiento por la pista podría ser un poco más rápida y quizás sea ésta la única característica que entorpezca un tanto el juego. Que en todo lo demás debe, sin duda, formar parte de una ludoteca que se precie.





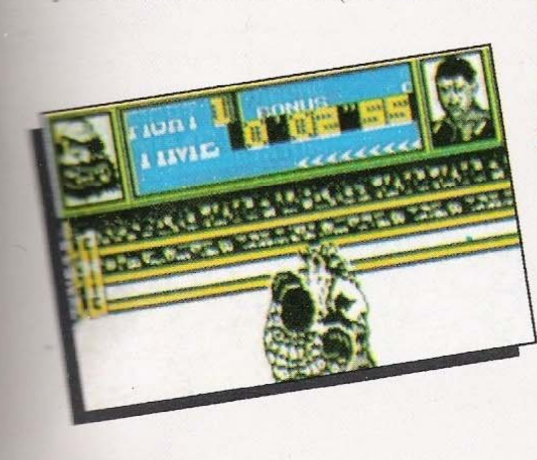


FRANK BRUNO'S BOXING

FRANK BRUNO'S BOXING fue uno de los programas que mayor expectación causaron en el público... Entre las características anunciadas se hallaban las posibilidades de ser contrincante de varios jugadores de todas las nacionalidades posibles. RAVIOLI era el representante italiano, BRUTOVSKI era el representante de la UNION SOVIETICA, etc... El juego no da pie a contemplaciones ni entrenamientos, así que se perfila el final de la carga, suena el gong y empieza la pelea. El primero de tus oponentes es un canadiense que parece muy duro de pelar: ojo con la cara porque te la puede poner a cuadros. En definitiva, no es tan rápido como podía parecer a primera vista y seguro que puedes con él. Las peleas se resuelven al cabo de tres minutos de combate sin ningún tipo de apelación, y hay tres combates. Estos se suceden uno tras otro hasta que alcances la cima del campeonato del mundo. Como te puedes imaginar, el nivel de dificultad sube con cada contrincante: FRENCHIE el francés y TRIBAL el africano (con el hueso en la nariz) son un auténtico hueso. Más hueso es, sin duda, el contrincante RAVIOLI MAFIOSI que, como su nombre indica, es un mafioso con cara de pocos amigos, y que lanza golpes diestros y todo lo sucios que se le puedan imaginar sobre la marcha. Cada jugador tiene un nombre asociado a la «facha» y nacionalidad.

Las posibilidades gráficas del ordenador están bien aprovechadas: verás en la parte superior izquierda y derecha las caras de los contrincantes: debajo de la pantalla se hallan los indicadores de fuerza de los oponentes y una vista parcial del ring (sólo uno de los lados de las cuerdas). El último de los contrincantes es PETER EL PERFECTO, que es, sin duda, mucho más fuerte que los contrincantes anteriores: deberás manejar a la perfección el *uppercut* y cuidarte de no perder uno solo de los descuidos de tu contrincante para atizarle. OJO !!! porque en cuanto te enganche verás que no te dejará salir vivo del cuadrilátero.

Lo único que se puede echar de menos en este programa es que se pierda gran parte de la visión de juego por el tan peculiar ángulo de visión que han seleccionado los programadores. Y el peculiar manejo a que obliga el juego desde los distintos ángulos.

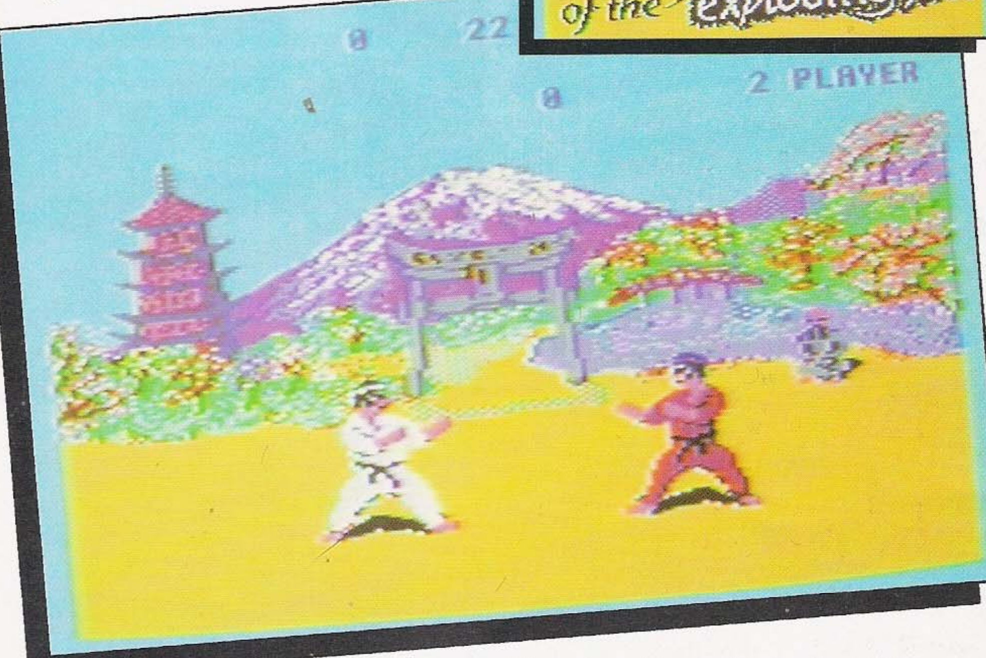




KARATE

BLACK BELT

BLACK BELT: cinturón negro. Sin duda, el primer programa que salió al mercado referente al tema que nos ocupa. ¡Los gráficos no eran lo que más resaltaba de este programa! Quizás por esto nunca llegó a aparecer en nuestro país. Tal vez la característica más relevante de este programa sea que permite crear una base de datos de características del jugador que tiene en cuenta todos los combates anteriormente realizados. Y como no todos somos iguales, podrás elegir el cinturón que corresponde a tu nivel de conocimientos o ambiciones.
El orden de los cinturones es el siguiente: blanco, amarillo, verde, azul, rojo y negro. Como el programador era consecuente, cada color se corresponde con un nivel de juego, y hay una serie de patadas que te serán imposibles en los niveles de conocimientos más bajos.
Concretamente, si quieres practicar el golpe que sale mucho en la película de KARATE que consiste en correr y saltar dando una patada, te será necesario correr el riesgo de coger el cinturón rojo.
Relacionado con el sistema de puntuación existe otra particularidad. Un árbitro en pantalla computa los tantos que se hacen. Esta puntuación te será otorgada según el grado de dificultad y la ejecución del golpe: así serán distintos un golpe fácil bien ejecutado que uno difícil ejecutado a medias. El programa dará puntos al contrincante que logre evitar el golpe.
El árbitro es muy estricto y no te dejará pasar ni una sola falta de los reglamentos.



Aparte de esto, quizás lo que más se eche de menos sea la animación, que es algo pobre, y mucho más el sonido, que recuerda muy vagamente al de los golpes.

EXPLODING FIST

No podía faltar en nuestro recorrido el EXPLODING FIST, el juego de estrellato indiscutible que lanzó a la fama para siempre a la MELBOURNE y que en ESPAÑA batió todo tipo de récords.
Lo más atractivo de EXPLODING es que en la época en que apareció supuso un extraordinario avance en la animación de los sprites. El desplazamiento de tu hombre es perfecto. Los pasos son lentos o rápidos a voluntad del joystick. Los movimientos son ultraprecisos y te permitirán realizar múltiples acrobacias y piruetas con el fin de que tu oscuro oponente vaya a moder el polvo.
El marcado carácter bélico de este juego queda patente cuando se sabe que de los 15 movimientos contemplados 11 son de golpes, y sólo 4 de dirección (derecha, izquierda, salto adelante y atrás). Todo esto conlleva un serio problema de manejo de palanca que podrás resolver optando por la opción PRACTICE del juego que te permitirá entrenarte hasta que te sientas ya capaz de enfrentarte con los tipos en la arena. Debes ser de lo más exquisito en el juego, ya que puedes correr el riesgo de que, aun ganando a tu adversarrio, el juez omnipresente te anule el combate por FULLERO. Para pasar de combate en combate deberás marcar tres tantos a tu adversario; el



marcador tiene vocación de oriental, ya que es la representación del símbolo del YING-YANG. Los combates siempre cambian de lugar y verás desfilar desde playas desiertas y templos deshabitados hasta salas de combate o entrenamiento.
El sonido tampoco estaba despreciado en este juego: cada golpe bien dado está recompensado con un grito inhumano de las «fauces» del televisor. ¡Superrealista!



THAI BOXING

THAI BOXING es uno de los últimos títulos que han salido al mercado de este tipo de juegos. Los THAI son luchadores que usan tanto técnicas dignas del más puro estilo de KARATE junto con técnicas propias del boxeo. En definitiva, se trata de una especie de FULL CONTACT: los luchadores llevan pantalones largos y guantes de boxeo. A decir de los que practican este deporte, es uno de los más duros que existen.

Son seis los niveles de dificultad que tiene este juego. Aquí todo lo superfluo está eliminado: no se puede cambiar el color del pantalón, ni tampoco los guantes; aquí no hace falta, ya que lo único importante es que empiece cuanto antes el combate. Los oponentes se distinguen por el color de los calzones. La pantalla de presentación recuerda algo a la del FRANK BRUNO'S BOXING: los dos contrincantes aparecen en la parte superior junto a los indicadores de fuerza o resistencia. Realmente resulta trágica esta forma de presentación, ya que a diferencia del Frank Bruno's, en el que los retratos sólo tienen una «dudosa» (por lo feos que son) función estética, aquí estos retratos son el auténtico indicador del estado de tu jugador y del oponente: verás que cuando te hayan «colocado» cuatro o cinco «galletas THAI» una fina gota de sangre bajará por tus mejillas. Conforme vaya habiendo más golpes y golpes, la cara se irá ensangrentando; parecerá que los ojos han desaparecido bajo los golpes y así hasta que tengas ganas de tirar la toalla... Pero esto no es ROCKY, y aquí no es posible: o ganas o pierdes. Y si ganas, la recompensa es que pasas a la siguiente fase de juego. Cambiando el port del joystick podrás optar por uno u otro jugador. Son, en total, 16 las acciones que contempla este juego. Están incluidas las patatas y puñetazos a todas las alturas, protecciones, etc... En suma, un excelente juego de combate que hará las delicias de los que amen los juegos rápidos, ya que éste es quizás el más rápido de todos los juegos de combate que os hemos presentado hasta ahora. Pero aún habrán más, no os creais!!!

THAI BOXING



NINJA



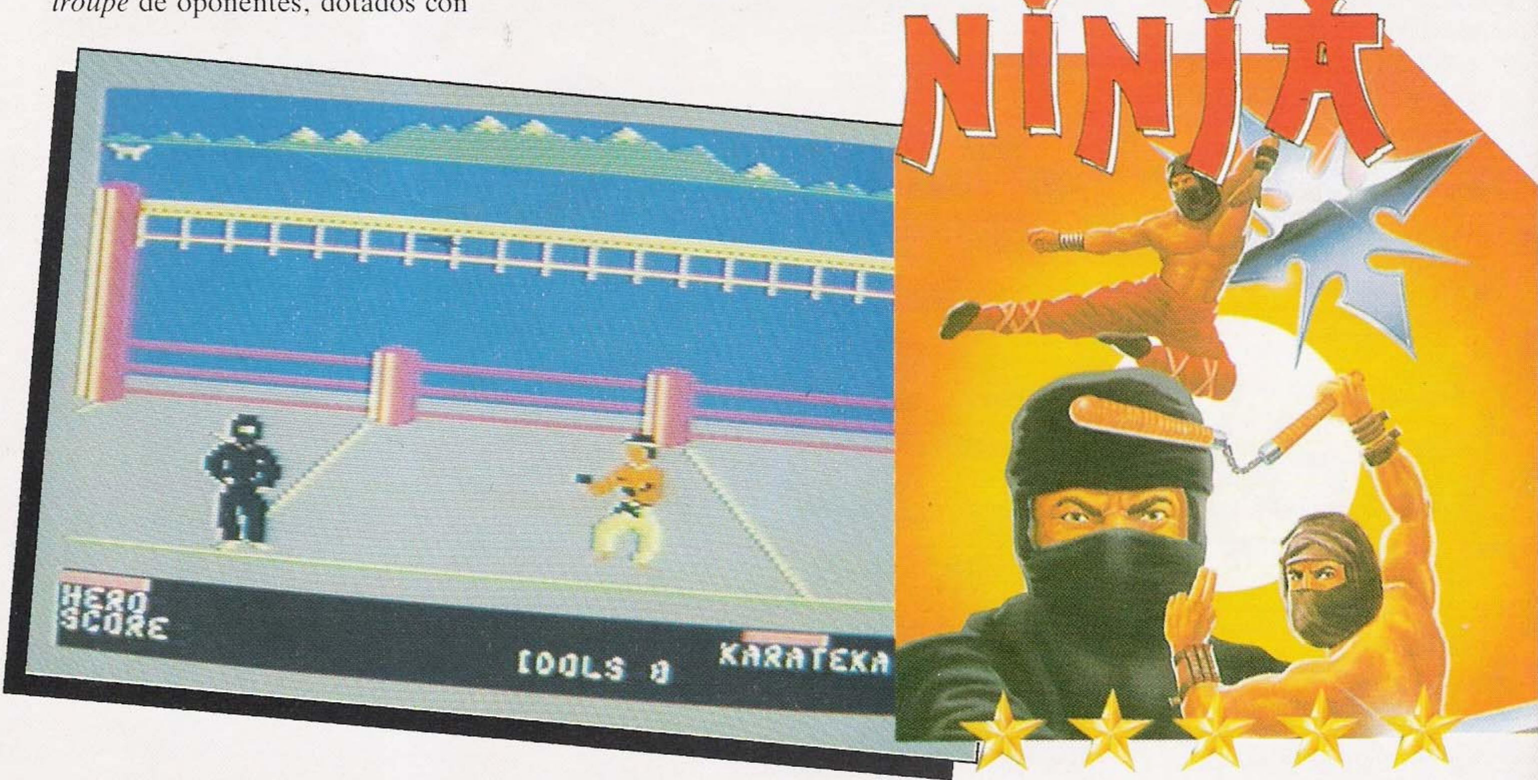
NINJA no podía faltar en nuestra selección de programas de lucha. Como siempre, se trata de salvar a la mujer de tus sueños (princesa oriental, por más señas), raptada por los pérfidos esbirros del mal, personificado este último por algún alto capitoste de la mafia asiática. Como muchos de los numerosos juegos de MASTERTRONIC, aunque a primera vista parezca un poco lento, posee una altísima adicción: las fases se suceden lo bastante rápido como para quedarse «enganchado» al joystick una «eternidad».

Tenemos en nuestro poder las armas típicas de los NINJA: nuestros puños de acero «inoxidable» para combates cuerpo a cuerpo, la espada y daga destructoras para segar cabezas de indeseables enemigos, y, por último, las letales *shaken*, estrellas portadoras de la muerte, para barrer y masacrar a distancia, que te ayudarán en tu periplo por los senderos luminosos de la venganza. Los recintos por los que tendremos que pasar son, en su totalidad, 20, de dificultad creciente y en las que nos encontramos con una inmensa *troupe* de oponentes, dotados con toda clase de armas y pericia.

En cuanto a los gráficos, **abundancia** y **profusión** son los adjetivos más adecuados para definirlos. El *scroll* es rapidísimo, aunque con ligeras imperfecciones en algunas partes del escenario, y los *sprites* impecables. El color tampoco deja nada que desear, y el sonido, modesto en comparación con los demás conceptos, es aceptable.

Teniendo en cuenta todo lo que hemos dicho, sólo cabe una conclusión: estamos ante un programa excelente.

Por otra parte, no os costará mucho haceros con él desde las primeras partidas, pues el nivel de dificultad no es muy alto, y además, las informaciones que aparecen en la pantalla os pueden dar en todo momento una idea exacta de vuestra situación.



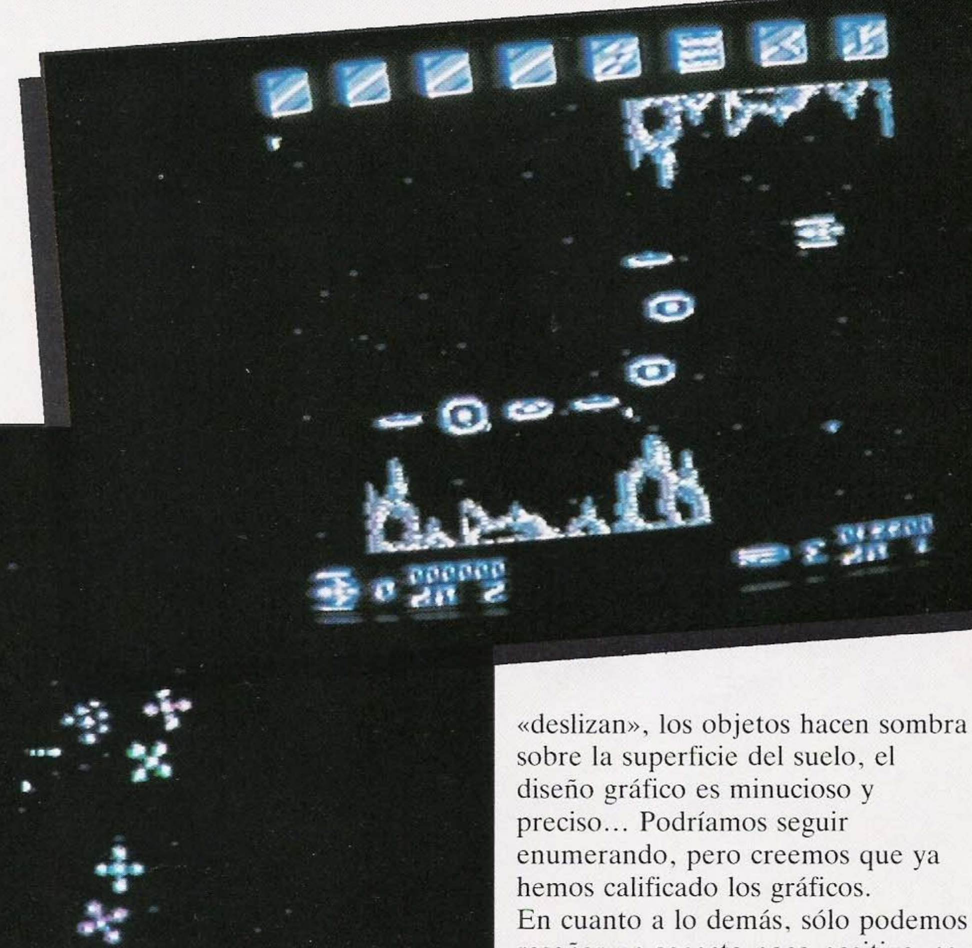
SOFTACTUALIDAD

CHAMPIONSHIP FOOTBALL

La casa es GAMESTAR, el programa tiene un antecedente extraordinario: el TWO ON TWO que tuvo la acogida que todos conocéis. Esta vez se trata de un fabuloso FOOTBALL AMERICANO, y aunque no sea nuestro deporte nacional bien vale la pena jugar a esta fabulosa emulación en la que parece que los jugadores salen de la pantalla cuando hacen un placage. Algunas jugadas son hasta ahora nunca vistas y los efectos gráficos son, simplemente, fenomenales.
Esperamos que la casa PROINSA, que distribuyó anteriormente el TWO ON TWO, comercialice este nuevo título.

GUNSHIP

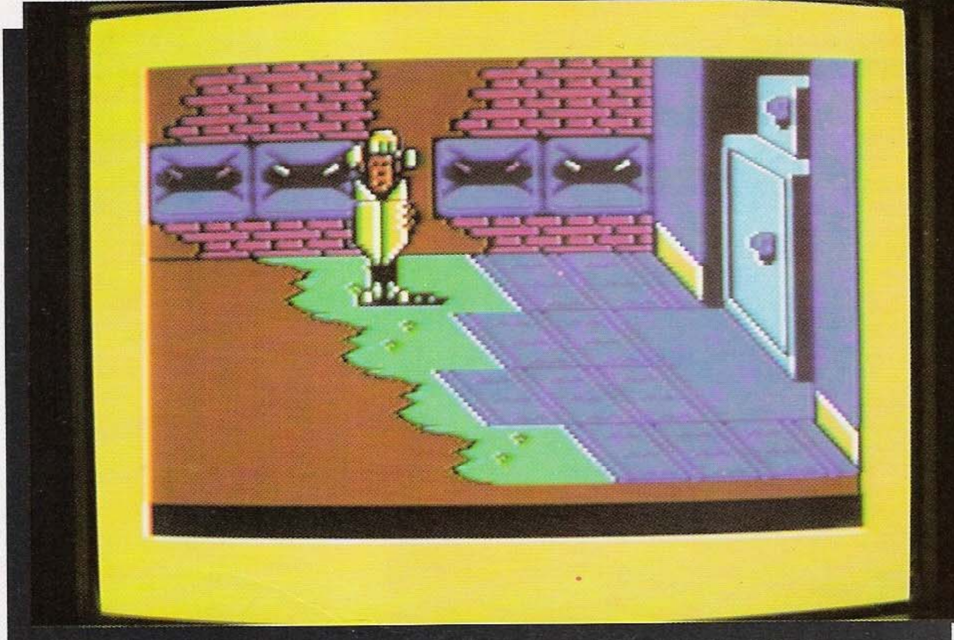
GUNSHIP es el último título de la prestigiosa casa MICROPROSE, especializada en todo tipo de simulaciones; se trata esta vez de pilotar el llamado APACHE, un fabuloso helicóptero dotado de unas características poco comunes y de una velocidad de vuelo y de una resistencia a toda prueba.
Como siempre, el manual de instrucciones es de lo más completo y la simulación, de un realismo extraordinario. Podrás seleccionar y modificar todos los parámetros, que incluyen desde los mapas, armamento de aparato... ¡hasta el color del gorro del piloto!
La visión es tridimensional y la velocidad de juego, realmente sorprendente.
El programa es de una calidad gráfica excepcional. Tiene todos esos pequeños detalles que a veces pasan desapercibidos, pero que entusiasman a quienes saben reparar en ellos. El *scroll* es rápido y suave, las figuras más que desplazarse se «deslizan», los objetos hacen sombra sobre la superficie del suelo, el diseño gráfico es minucioso y preciso... Podríamos seguir enumerando, pero creemos que ya hemos calificado los gráficos.
En cuanto a lo demás, sólo podemos reseñar un aspecto poco positivo: se trata del argumento, excesivamente simplista, y del desarrollo de la acción, algo monótono.



KILLED UNTIL DEAD

Este es el último título de la prestigiosa casa norteamericana ACCOLADE, autora, entre otros, del famoso ACE OF ACES. Esta vez se trata de resolver un misterio tipo AGATHA CHRISTIE. Un asesinato, sospechosos, herencias, lugares insólitos y peligros innumerables y, ¡cómo no!, un final que, sin duda alguna, no desmerece de esta fabulosa casa de software que nos tiene acostumbrados a una calidad gráfica sin par.
La acción transcurre en una serie de pantallas de desarrollo tridimensional que, aunque se parecen mucho entre sí, siempre encierran peligros diferentes.
El nivel de dificultad es quizás demasiado alto, pero esto, lejos de restar interés al programa, contribuye a conseguir que no nos podamos separar de la pantalla hasta haber logrado algún pequeño éxito. No obstante, es muy probable que nos desesperemos un poco al principio. Es cuestión de voluntad. Al final de cada partida, podremos seguir nuestros progresos comprobando el nivel alcanzado en la tasa de «credibilidad del héroe», expresada en tantos por ciento. No está de más que te recordemos que la escritora británica Agatha Christie (1891-1976) constituye una de las cumbres de la novela policiaca, habiendo creado títulos tan estimulantes como clásicos como «Diez negritos», «El asesinato de Rogelio Ackroyd» y «Asesinato en el Orient Express», de los cuales se han hecho acertadas y lujosas adaptaciones cinematográficas que, seguro, habréis visto en la pantalla grande o disfrutado en la intimidad de vuestro hogar a través del video o de la lectura.





La principal innovación que introduce este juego, pues la originalidad es uno de los factores que más hemos de valorar, es que las distintas pantallas de que se compone son interdependientes entre sí, es decir, no se trata de «habitaciones» que simplemente se comunican sin continuidad clara, separadas por tabiques, sino de un solo plano contínuo sin partes aisladas.
Por lo demás, el juego es un derroche de imaginación y de colorido que nos hará pasar muy buenos ratos sin que en ningún momento lleguemos a aburrirnos.

INSPECTOR GADGET

INSPECTOR GADGET es, sin duda, uno de los títulos que más expectativas ha creado en el mercado. La versión en nuestro micro es ligeramente posterior a la de otras máquinas pero, a juzgar por las fotografías, merecía la espera: para los que están hartos de ver cómo otro usa los GADGETOPATINES, o bien el GADGETOBRAZO llegó el juego (sin duda único), que os permitirá tener toda la panoplia del famoso héroe televisivo,

FEUD

• MASTERTRONIC ■ JUEGO



El mes pasado os hablamos en el apartado de actualidad de un fenomenal programa que este mes ha sido distribuido por DRO-SOFT en nuestro país. El programa están en la línea de precio de MASTERTRONIC, aunque la casa que lo creó originalmente es BULLDOG. El argumento es relativamente sencillo: en la edad media dos magos (uno bueno y otro malo) pugnan por el control de una zona. La pugna se hace a base de numerosos hechizos.

Los hechizos y conjuros son bastante fáciles de realizar, ya que basta acercarse al caldero, seleccionar de la lista de los 11 embrujamientos el que se quiere y, acto seguido, tomar los ingredientes del libro.
Los gráficos son quizás lo mejor del juego, aunque la animación no se queda corta, ya que este programa logra gran parte de su realismo por la elección de los colores tanto de los personajes como de los escenarios.
El juego está muy bien ambientado y ha debido ser fruto de un laborioso trabajo: prueba de ello es que los autores han respetado la vestimenta de la época.
El mago que tratará de eliminarte es muy rápido, así que deberás ser veloz, sobre todo en lo que a búsqueda de ingredientes se refiere. Todos estos son vegetales que deberás coger y cortar con una hoz y mezclar con otros para crear conjuros.

ANIMACION	10
INTERES	8
GRAFICOS	10
COLOR	10
SONIDO	7
TOTAL	45



EL ZOCO



Vendo Interface copiador de cassette a cassette (copia con dos unidades lectoras C2N) de uso personal en 3.000 ptas. Vendo los programas originales con sus instrucciones y cajas: Express Rider, Two on two, Skooldaze, Broad Street, Decathlon, etc. Todos ellos en cinta a 400 ptas. c/u para C-64. José R. Díaz Glez. C/ Río Requejada, 2, 1.° izq. 33460 Llaranes. Asturias. Tl: (985) 57 28 31.

Necesito las instrucciones en castellano de los siguientes juegos: Back to the Future, Eidolon, Mission Elevator, Strike force Cobra, Jack the Nipper, Saboteur, Master of Lamps. Si tienes alguna escríbeme. Te las cambiaré por programas. Tengo casi 200. También vendo e intercambio. Juan Manuel Mínguez. C/ Independencia, 10, 2.° derch. 40002 Segovia.

Vendo Commodore 128, joystick, 30 revistas, programas, manuales y libros por 50.000 ptas. y por 65.000 incluyendo DBASE II, Wordstar, Mbasic, Cobol, etc. Junto o por separado. Perfecto estado. José Marsá Mallol. Tl: 352 98 90 de 2 a 4 h. Barcelona

---

Intercambio programas en cinta para el C-64. Dispongo de más de 400 títulos algunos de ellos últimas novedades. Prometo contestar a todos. Domingo Saura Ballester. C/ Exportador Ocampos, 3. Pozo Estrecho. Cartagena. Murcia. Tl: (968) 55 64 57.

Cambio juegos para el C-64. Tengo parte de las últimas novedades. Mándame tu lista. Pedro Lanchares Amores. C/ Avenida de Portugal; 16, 3.° B. 24400 Ponferrada. León.

Cambio juegos e instrucciones en cinta para el C-64. Tengo unos 450 de gran calidad. Escribe aunque tengas pocos. Óscar Orallo. Plza. La Fortaleza, 11, 4.° C. 24400 Ponferrada. León.

Vendo ordenador Dragon 200, un monitor color, 15 juegos, precio a convenir. Mercedes. Tel: (93) 230 40 57.

Cambio, compro o vendo programas para el C-64, también me gustaría ponerme en contacto con clubs informáticos. Iñaki Jiménez Carcavilla. C/ J. A. Fdez., 6, 6°, D. 31500 Tudela. Navarra. Tl: (948) 82 15 73.

Vendo impresora CPA-80 con interface Commodore, con ocho meses, casi sin usar y maletín de papel continuo por 45.000 ptas. José Fco. Espinosa Jurado. C/ Alfonso de Palencia, 11, 5.°, D. 29007 Málaga. Tl: (952) 27 92 21.

Vendo 5 juegos originales para el C-64 (Summer Games II, Sky Fox, Sanxion, Two on Two, etc.). Baratísimos. Cambio juegos novedades. Pedro Gómez González. C/ Renedo, 14 1.° D. 47005 Valladolid. Tl: 21 13 82.

Intercambio/vendo/compro juegos para el C-64. Poseo más de 500 todos de una gran calidad. La mayoría con turbo. Juegos como: Uridium, Commando, Green Beret, Rambo, Fist II, Int. Karate I y II, etc. David Tapias Monné. Sardá i Cailá, 7, 4.°-2.ª. Reus. Tarragona. Tl: 31 66 17/31 95 94 (a partir de las 20.30).

Me gustaría ponerme en contacto con personas que tengan un C-64-128 de La Coruña, para intercambio de revistas, programas y trucos. Elia M.ª Domínguez. C/ Arenal, 29, 2.°. 15002 La Coruña.

Urge vender C-64K a estrenar con manuales, juegos, joystick y DATACASSETTE. Todo ello por 40.000 ptas. Llamar a partir de las 18 h. Tl: (91) 250 18 57. Juan Pablo Fdez. Conejo. P.° de la Habana, 190, 2.°, A. 28036 Madrid.

Ofrezco 430 programas a cambio de una unidad de disco, también daría unas 6.000 ptas. Intercambio programas para el C-64, tengo últimas novedades como Express Raider, Arkanoid, Skate Rock, etc. Asimismo me urge conseguir los programas Dasic Lightning y Game Marker los compraría o cambiaría por juegos. Fco. Javier Morante López. C/ Pompeu Fabra, 65, ático 1.ª. 08922. Santa Coloma de Gramanet. Barcelona.

Cambio programas para C-64, en cinta o disco, poseo últimas novedades. José Luis Rodríguez. C/ Escultor Ordóñez, 15-23, 2-3 esc. A. 08016 Barcelona. Tl: 340 39 27.

Vendo C-64 (36.000 ptas.), Datassette (7.000 ptas.), F. Cartridge (8.500 ptas.), L. M. para avanzados, L. M. para Commodore, 54 interno, Diccionario 64, Rutinas 64, Guía del usuario, Lenguaje máquina, Gráficos y sonido, Programación en lenguaje Pascal y algunos más (11.000 ptas.) y otras cosas más. Todo más juegos y utilidades (69.000 ptas.). Josep Many. Canovelles. Barcelona. Tl: 849 38 52.

Vendo Commodore C-128,1571 Disk drive, Riteman C+ y monitor color Dynadata 40/80 columnas con cables incluidos, todo en conjunto o por lotes, a parte de numerosos programas tanto en C-64, C-128 y CP/M mode: Marco A. Zamit Engo. Avda. País Valenciá, 45. 46850 Olleria. Valencia. Tl: (96) 220 01 16 sólo mañanas.

Vendo C-64 más unidad de disco 1541, Impresora MPS 802, cassette C2N, Lápiz óptico, joystick. Vendo todo con procesador de textos, hoja de cálculo, Base de datos y programas gráficos... Además regalo muchos programas de juegos y utilidades, libros y revistas. Todo por 110.000 ptas. Santiago García. Madrid. Tl: (91) 202 32 01 de 14 a 15 h y de 20.45 a 23.45 h.

Vendo para el C-128 Compiladores de Cobol, lenguaje C y superbase para C-128. Vendo compiladores de Cobol, lenguaje C, Ada, Pascal y superbase (ninguno de los programas necesita usar CP/M.) Cobol o C a 25.000 ptas. Fernando J. Gallardo Romero. C/ Vetones, 9, 1.° Apdo. 315. 06800 Mérida. Badajoz. Tl: 31 07 92.

Intercambio/vendo juegos tipo Cauldron II, Deactivators, Fist II, Titanic y casi 400 por el estilo. También vendo interfaces copiadores de cintas a 3.000 cucas negociables. Contestaré a todos. Antonio Torrente. Polígono Cañelles B1 D-8, 7.°-3.ª. 08033 Barcelona. Tl: (93) 427 47 99.

Club para el C-64 CAMBIAMOS/VENDEMOS utilidades, programas, etc. Tenemos últimas novedades: Enduro Racer, Gauntlet II, Arkanoid, etc. Admitimos nuevos socios. Animaos. José Luis Moreno Córdoba. C/ Rey, 89, 3.°, A. 28300 Aranjuez. Madrid.

Club de usuarios de Spectrum, Commodore. MSX e IBM hace ampliación de socios. Para poder participar escribe: KIMBO YAÑEZ. C/ Conca de Tremp, 83, 1.°-1.ª. 08032 Barcelona. Porfa, envía un sello.


TU ERES FANTASTICO
¡Unete a
LOS 4 FANTÁSTICOS!
AL ESTILO COWBOY
125
LOS 4 FANTÁSTICOS
Nº 39
125
LOS 4 FANTÁSTICOS
¡ATENCION! ¡EL MONSTRUO ATACA DE NUEVO!
¡ANTORCHA, ESPERA! ¡AHORA ME TOCA A MI!
Nº 42
125
PTAS
CADA MES EN TU QUIOSCO
COMICS forum

CINCO MINUTOS ANTES DE COMPRAR UN JUEGO A 875 Ptas.

ECHALE UN VISTAZO A ESTOS JUEGOS DE 875 Ptas.



875 Ptas.
VERSION CASSETTE

COBRA
SOFTWARE

SÍGUENOS EL JUEGO.

ZAFIRO SOFTWARE DIVISION Paseo de la Castellana, 141. 28046 Madrid
Tel. 459 30 04. Tel. Barna. 209 33 65. Telex: 22690 ZAFIR E